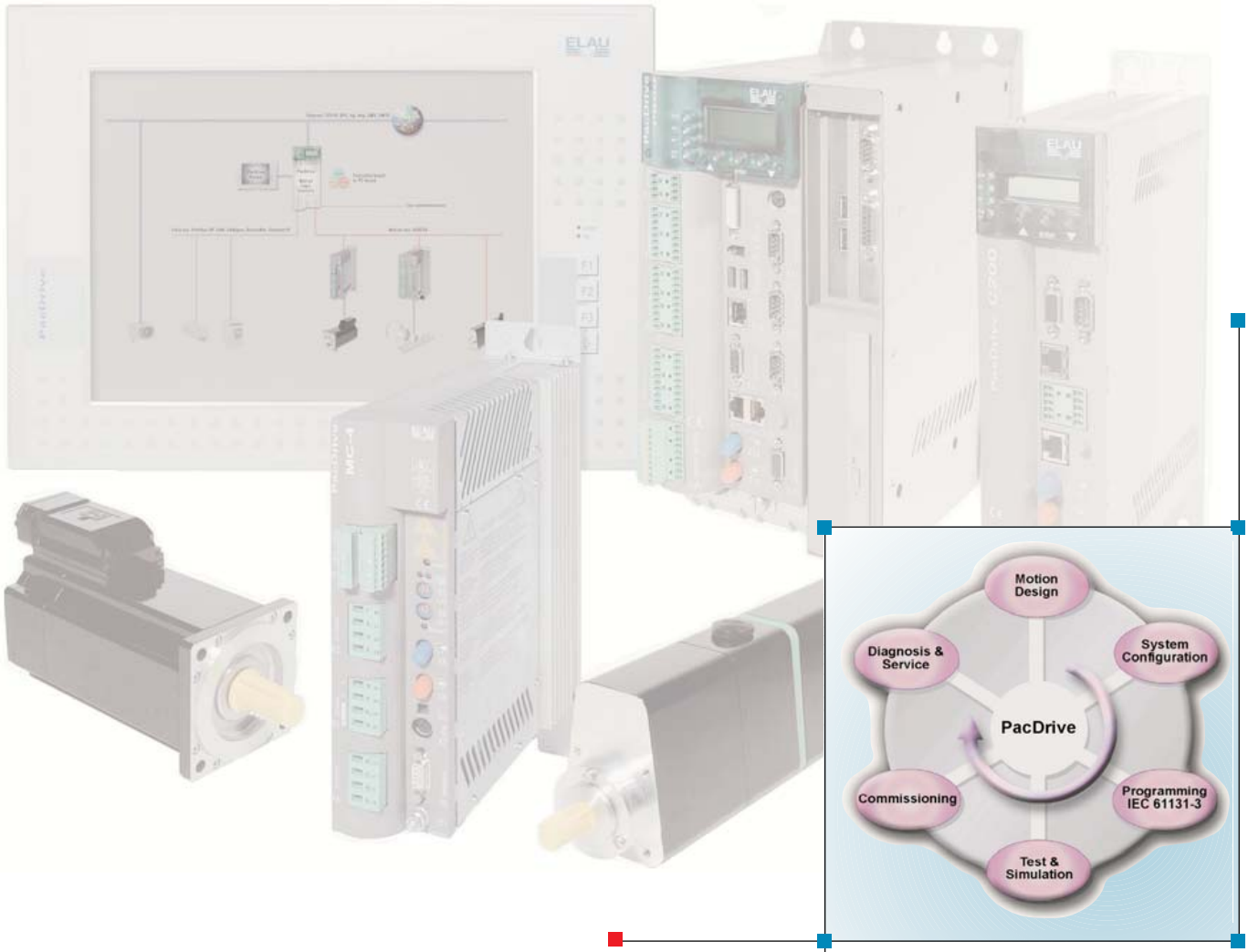
Manuel de programmation

# Système PacDrive™




PDM_UserMan_fr1003
Numéro d'article : 17130061-003
Edition : 10.2003

## Copyrights

© Tous droits réservés à la société ELAU AG, également dans le cas de dépôts de droits de protection.

Sans l'accord par écrit de la société ELAU AG, il est interdit de reproduire, transmettre, réécrire, sauvegarder sur des supports de données, ainsi que de traduire dans une autre langue, respectivement dans une autre langue de programmation, même partiellement, la documentation présente ainsi que le logiciel et les micro programmes s'y rapportant.

Toutes les mesures possibles ont été prises afin de garantir l'exactitude de la documentation présente du produit. Cependant, étant donné les améliorations constantes, effectuées en permanence, en ce qui concerne le matériel et le logiciel, la société ELAU AG ne peut ni en garantir l'intégrité ni l'exactitude.

## Marques déposées

PacDrive est une marque déposée de la société ELAU AG.

Toutes les autres marques déposées citées dans cette documentation sont la propriété exclusive de leurs fabricants.

ELAU AG
Dillberg 12
D-97828 Marktheidenfeld

Tél. : +49 (0)9391/606-0
Fax : +49 (0)9391/606-300

eMail : info@elau.de
Internet : www.elau.de

PDM_UserMan_00_fr.fm

# Table des matières


1 Préface 9

1.1 Introduction ... 9
1.2 Symboles, signes et formes de représentation ... 10

2 Conditions de sécurité générales 11

2.1 Généralités ... 11
2.2 Représentation des consignes de sécurité ... 12
2.3 Utilisation conforme aux prescriptions ... 13
2.4 Sélection et qualification du personnel ... 14
2.5 Risques subsistants ... 14
2.5.1 Montage et manipulation ... 15
2.5.2 Contact avec des pièces électriques ... 16
2.5.3 Très basses tensions à séparation sûre ... 17
2.5.4 Mouvements dangereux ... 18

3 Vue d'ensemble du système 19

3.1 Concepts d’entraînement pour les machines d’emballage ... 19
3.2 Structure du système d’automatisation PacDrive™ ... 21
3.3 Concept ... 22
3.4 Composants ... 23

4 Bases de la norme IEC 61131 25

4.1 Automates programmables ... 25
4.2 IEC 61131 ... 28
4.2.1 Le modèle de programmation ... 29
4.2.2 Le modèle de communication ... 31
4.2.3 Unités d’organisation de programme UOP ... 43
4.2.4 Les langages de programmation ... 49
4.2.5 Opérateurs ... 71
4.2.6 Opérandes ... 86
4.2.7 Bibliographie ... 91
4.3 Particularités du PacDrive™ ... 92
4.3.1 Principe de fonctionnement ... 92
4.3.2 Types de données de la configuration de commande ... 93
4.3.3 Taille du programme et des plages de variables ... 93

5 Recommandations pour la programmation 95

5.1 La notion de mappage ... 95


PDM_UserMan_frVZ.fm

5.1.1 Structure des machines d'emballage ..... 95
5.1.2 Le système d'automatisation ..... 96
5.1.3 De la définition du problème à la solution ..... 97
5.1.4 Méthode de solution du problème ..... 98
5.2 Structure d'un projet ..... 99
5.2.1 Organisation d'un programme par tâches ..... 99
5.3 Structure du programme ELAU ..... 109
5.3.1 Fonctions et blocs fonctionnels d'ELAU ..... 112
5.3.2 Transmission des paramètres ..... 113
5.4 Identificateurs ..... 114
5.5 Majuscules/minuscules ..... 116
5.6 Signes autorisés ..... 116
5.7 Réalisation d'un projet PacDrive™ ..... 116
5.7.1 Définition du problème ..... 116
5.7.2 Matériel ..... 118
5.7.3 Configuration ..... 119
5.7.4 Programme utilisateur ..... 122
5.7.5 Observer & Piloter ..... 128
5.7.6 Documentation et sauvegarde d'un projet ..... 130
5.8 Elargissement de l'exemple par des fonctions de mouvement ..... 131
5.8.1 Exposé du problème ..... 131
5.8.2 Matériel ..... 133
5.8.3 Configuration ..... 134
5.8.4 Programme utilisateur ..... 136
5.8.5 Observer & Piloter ..... 138
5.8.6 Documentation et sauvegarde d'un projet ..... 142

6 Stratégies de recherche d'erreurs 143

6.1 Marche à suivre ..... 143
6.2 DELs ..... 144
6.3 Configuration de commande ..... 145
6.4 Enregistreur de messages ..... 146
6.4.1 Fonction ..... 146
6.4.2 Les fonctions individuels de l'enregistreur de messages ..... 147
6.4.3 Filtrage des messages ..... 149
6.4.4 Etablir la liaison avec le PacDrive Controller MAx-4 en cas d'erreur ..... 150
6.5 Autres moyens de diagnostic ..... 152
6.5.1 Trace ..... 152
6.5.2 Mise au point et test ..... 152
6.5.3 La bibliothèque Check ..... 152

PDM_UserMan_frVZ.fm

6.5.4 Hiérarchie d'appels (CallStack) ... 153

7 Télémaintenance 155

7.1 Sommaire ... 155
7.2 Connexion à distance au PacDrive Controller MAx-4 ... 157
7.2.1 Principe de fonctionnement ... 157
7.2.2 Choix du modem ... 160
7.2.3 Paramètres du projet pour le PacDrive Controller MAx-4 ... 162
7.2.4 Configuration de l'ordinateur distant ... 167
7.2.5 Etablissement de la connexion entre le PacDrive Controller et l'ordinateur distant ... 191
7.2.6 Recherche/localisation de défauts ... 192
7.3 Connexion à distance au PacDrive Controller via l'ordinateur distant ... 197
7.3.1 Principe de fonctionnement ... 197
7.4 Connexion à distance avec plusieurs automates PacDrive Controller via le routeur ... 198
7.4.1 Principe de fonctionnement ... 198
7.5 Connexion via Internet ... 199
7.5.1 Connexion directe via une adresse IP ... 199
7.5.2 Connexion VPN (Virtual Private Network) ... 200
7.6 Accès au PacDrive Controller via un navigateur classique (non disponible) ... 201
7.6.1 Principe de fonctionnement ... 201

8 Réseaux 203

8.1 Concept de base ... 203
8.1.1 Peer to Peer ... 203
8.1.2 Client-Serveur ... 204
8.2 Topologies ... 205
8.2.1 Les topologies ... 205
8.2.2 Topologie en bus ... 205
8.2.3 Topologie en étoile ... 206
8.2.4 Topologie en anneau ... 207
8.2.5 Topologies hydrides ... 208
8.2.6 Supports de transmission ... 210
8.3 Adresse MAC ... 211
8.4 Protocoles d'accès ... 212
8.4.1 Le protocole CSMA/CD avec détection de collision ... 212
8.4.2 Token Passing ou Passage du jeton ... 213
8.5 Les systèmes d'exploitation de réseau ... 214
8.5.1 Netware de Novell ... 214
8.5.2 Windows NT et Windows 2000 de Microsoft ... 214
8.6 Le protocole TCP/IP ... 214

8.6.1 Comparaison des modèles de référence ... 214
8.6.2 Principaux protocoles ... 215
8.6.3 Adressage IP ... 217
8.6.4 Masque de sous-réseau (subnet mask) ... 217
8.6.5 Classes d'adresses IP ... 218
8.6.6 Réseaux privés IP ... 218
8.6.7 Affectation d'adresse IP ... 219
8.6.8 Résolution des noms ... 220
8.6.9 Résolution de noms NetBIOS et IP Host ... 220
8.7 Interconnexion de réseaux ... 221
8.7.1 Les équipements d'interconnexion ... 221
8.7.2 Outils de gestion des problèmes TCP/IP ... 224

9 Caractéristiques du système 231

10 Identification de version et compatibilité 233
10.1 Identification générale de version ... 233
10.2 Compatibilité EPAS-4 - MAx-4 ... 233
10.3 Compatibilité MAx-4 - MC-4 ... 233
10.4 Compatibilité MAx-4 - Bibliothèques IEC (IEC-Libraries) ... 233

11 Diagrammes des temps – drive «MotorController» 235
11.1 Mise en marche de la validation ... 235
11.2 Arrêt de la validation ... 236
11.2.1 Décélération avant expiration du temps de décélération maximal ... 236
11.2.2 Arrêt réglé de l'entraînement après dépassement du temps de décélération maximal 237
11.2.3 Arrêt non réglé de l'entraînement après dépassement du temps de décélération maximal 238
11.3 La réaction d'erreur « B » ... 239
11.3.1 Décélération avant expiration du temps de décélération maximal ... 239
11.3.2 Dépassement du temps de décélération maximal ... 240
11.4 La réaction d'erreur « A » ... 242
11.5 Arrêt par suite de surcharge (Overload) ... 243
11.5.1 Arrêt par suite de surcharge avant expiration du temps d'arrêt maximal ... 243
11.5.2 Arrêt par suite de surcharge avec dépassement du temps d'arrêt maximal ... 245

12 ENI-4 247
12.1 Fonction de l'interface d'ingénierie ENI ... 247
12.1.1 Mode multiutilisateur ... 248
12.1.2 Gestion de versions ... 248
12.1.3 Accès de clients externes ... 248

PDM_UserMan_frlVZ.fm

12.2 Structure et communications de l'interface ENI ..... 249
12.3 Structure interne du système de stockage de données/L'explorateur ENI ..... 250
12.4 ENI en tant que composante d'EPAS-4 ..... 251
12.5 ENI Admin (Gestionnaire ENI) et ENI Control (Contrôleur ENI) ..... 254

13 FAQ (Foire Aux Questions) 257

14 Glossaire 259

15 Annexe 285
15.1 Adresses de contact ..... 285
15.2 Documentation supplémentaire ..... 286
15.3 Formations produit ..... 288
15.4 Modifications ..... 289
15.5 Stichwortverzeichnis ..... 291
15.6 Formulaire de rapport de défaut ..... 297

# 1 Préface

## 1.1 Introduction

Prière de lire ce manuel attentivement avant la première utilisation des composants ELAU.

Veillez en particulier à respecter les consignes de sécurité figurant au chapitre 2.

Seules les personnes répondant aux exigences de qualification énumérées au chapitre 2.4 sont autorisées à effectuer des travaux sur les composants ELAU.

Un exemplaire du manuel doit en permanence être à la disposition du personnel chargé des travaux sur les composants ELAU.

Le manuel vous aide à utiliser les composants de manière correcte et en toute sécurité et à exploiter leurs possibilités en conformité avec les prescriptions.

Prière de respecter ce manuel. Ainsi vous contribuez à éviter des risques, à réduire les coûts de réparation et les temps d’arrêt et à augmenter la durée de vie et la fiabilité des composants.

Vous devez également vous conformer aux règles et dispositions relatives à la prévention des accidents et à la protection de l’environnement en vigueur dans le pays d’utilisation et sur le lieu d’implantation.

PD_VerwendStd_fr.fm

## 1.2 Symboles, signes et formes de représentation

La présente documentation utilise les symboles et signes suivants :

| Représentation | Signification |
|---|---|
| ■ | Signe d’énumération de premier niveau |
| – | Signe d’énumération de deuxième niveau |
| ▸ | **Symbole d’action** : le texte suivant ce symbole indique une action à exécuter. Exécutez les actions dans l’ordre indiqué. |
| ✓ | **Symbole de résultat** : le texte suivant ce symbole indique le résultat d’une action. |
| *Italiques* | Les termes techniques (*Paramètres* , par exemple) sont signalés dans le texte descriptif par des italiques. |
| `Police avec sérifs` | Les codes de programme éventuellement contenus dans ce guide sont imprimés avec une police avec sérifs. |
|  | **Symbole d’information** : vous trouverez après ce symbole des indications utiles ou des astuces concernant l’utilisation des composants. |
| | **Symbole de mise en garde** : ce symbole attire votre attention sur des indications importantes concernant la sécurité. |
| | Vous trouverez après ce symbole des informations sur le contenu du chapitre pour vous faciliter l'orientation. |

*Tableau 1-1: Symboles, signes et formes de représentation*

PD_VerwendStd_fr.fm

# 2 Conditions de sécurité générales

Ce chapitre contient des consignes générales destinées à garantir la sécurité lors de l'intégration. Toute personne utilisant des composants ELAU ou chargée de travaux là-dessus est tenue de lire et de respecter ces consignes de sécurité.

Des indications concrètes sont données au fil des pages à chaque fois qu'une intervention comporte des risques. Elles contiennent une description du danger éventuel et des mesures de prévention à mettre en œuvre.

## 2.1 Généralités

Les composants ELAU sont réalisés selon les règles de l'art et les prescriptions de sécurité en vigueur. Leur utilisation peut cependant provoquer de graves dommages corporels et matériels, voire entraîner la mort, si :

- vous n'utilisez pas les composants conformément aux prescriptions,
- les travaux sur les composants ne sont pas exécutés par des spécialistes ou des personnes spécialement formées,
- vous modifiez ou transformez les composants de manière inadéquate,
- vous omettez, au terme de l'installation, de la mise en service ou de la réparation, de tester les mesures de protection mises en œuvre,
- vous ne respectez pas les consignes et les prescriptions de sécurité.

Les composants ne doivent être exploités que dans un état technique parfait, dans le respect des prescriptions et en pleine connaissance des dangers engendrés et des règles de sécurité à observer.

La sécurité de fonctionnement des composants est également déterminée par le soin apporté au transport, au stockage, au montage, à l'installation et à la maintenance.

En cas d'apparition de situations susceptibles de compromettre la sécurité ou d'entraîner des modifications du comportement en service, il convient d'arrêter immédiatement le(s) composant(s) et de prévenir le personnel de service compétent.

Veillez par ailleurs à respecter :

- les indications figurant sur les étiquettes d'interdiction, d'obligation ou de mise en garde apposées sur le composant, sur les composants raccordés et dans l'armoire,
- les lois et décrets applicables,
- les notices d'utilisation des autres composants,
- les prescriptions de sécurité et de prévention des accidents locales et nationales applicables.

## 2.2 Représentation des consignes de sécurité

### Classes de danger

Les consignes de sécurité contenues dans ce guide sont classées en différentes catégories. Le tableau ci-dessous vous indique la signification du symbole (pictogramme) et des mots d'avertissement qui l'accompagnent.

| Pictogramme | Mot d'avertissement | Définition |
|---|---|---|
| | **DANGER !** | Attire l'attention sur une situation directement dangereuse pouvant entraîner la mort ou des blessures graves en cas de non-respect des prescriptions de sécurité. |
| | **AVERTISSEMENT !** | Attire l'attention sur une situation potentiellement dangereuse pouvant entraîner des blessures graves ou des dommages matériels importants en cas de non-respect des prescriptions de sécurité. |
| | **ATTENTION !** | Attire l'attention sur une situation potentiellement dangereuse pouvant entraîner des dommages matériels en cas de non-respect des prescriptions de sécurité. |

PD_SicherhMax_fr.fm

## 2.3 Utilisation conforme aux prescriptions

Les composants ELAU sont destinés à être montés sur une machine/installation ou associés à d'autres composants pour former une machine ou une installation. Il ne doivent être utilisés que dans les conditions d'installation et d'exploitation décrites dans la documentation. Vous devez impérativement utiliser les accessoires et auxiliaires (composants, câbles, conducteurs, etc.) spécifiés dans la documentation. Les appareils ou composants d'autres constructeurs qui ne sont pas expressément agréés par ELAU ne doivent pas être utilisés.

Pour garantir une utilisation conforme aux prescriptions, vous devez en outre :

- respecter les guides d'utilisation et autres documentations (voir annexe),
- observer les consignes d'inspection et de maintenance.

***Utilisation non-conforme aux prescriptions***

Il convient de vérifier et de respecter les conditions de service sur le lieu d'implantation en se référant aux caractéristiques techniques indiquées (indications de puissance et conditions d'environnement).

Toute mise en service est interdite tant que l'on n'a pas la certitude que la machine ou l'installation dans laquelle le(s) composant(s) est (sont) monté(s) est conforme, dans son ensemble, aux exigences de la directive machines 98/37/CEE.

Il est en outre nécessaire de respecter les normes, directives et prescriptions suivantes :

- EN 60204 Sécurité des machines :
  Equipement électrique des machines industrielles.
- EN 292 parties 1 et 2 Sécurité des machines :
  Notions fondamentales, principes généraux de conception.
- EN 50178 Equipement électronique utilisé dans les installations de puissance.
- Directive CEM 89/336/CEE

## 2.4 Sélection et qualification du personnel

Ce guide s’adresse exclusivement à un personnel technique qualifié possédant des connaissances approfondies dans le domaine des automatismes.

Seul un personnel qualifié est à même de comprendre le sens des consignes de sécurité et de les mettre correctement en œuvre.

Ce guide est essentiellement destiné aux constructeurs et aux utilisateurs des domaines de la construction de machines et de l’électrotechnique ainsi qu’aux programmateurs et ingénieurs de maintenance et de mise en service.

***Travaux sur les équipements électriques***

Les travaux sur des équipements électriques ne doivent être effectués que par des électriciens qualifiés ou par des personnes spécialement formées placées sous la direction et la surveillance d’un électricien, en conformité avec les règles électrotechniques.

Un électricien qualifié est une personne qui, par sa formation, son savoir, son expérience et sa connaissance des dispositions applicables, est en mesure de :

- juger le travail qui lui est confié,
- identifier les dangers potentiels,
- prendre les mesures de sécurité qui s’imposent.

## 2.5 Risques subsistants

Les risques pour la santé des personnes engendrés par les composants ELAU ont été minimisés par la mise en œuvre de mesures constructives ou de techniques de sécurité. Il existe cependant des risques subsistants liés au fait que les composants travaillent avec des tensions et des courants électriques.

## 2.5.1 Montage et manipulation

**AVERTISSEMENT !**

Risques de blessures lors de la manipulation !
Blessures par écrasement, cisaillement, coupure et choc !

- Respecter les prescriptions générales d'installation et de sécurité lors de la manipulation et du montage.
- Utiliser des dispositifs de montage et de transport appropriés. Recourir, si nécessaire, à des outils spéciaux.
- Prévenir les risques de pincement et d'écrasement par des mesures appropriées.
- Porter, si nécessaire, des vêtements de protection appropriés (comme des lunettes de protection, des chaussures de sécurité ou des gants isolants).
- Ne pas stationner sous des charges suspendues.
- Nettoyer immédiatement les liquides répandus sur le sol afin de prévenir les risques de glissade.

## 2.5.2 Contact avec des pièces électriques

Le contact avec des pièces soumises à des tensions supérieures à 50 V peut être dangereux pour les personnes. Lors du fonctionnement des appareils électriques, certaines pièces sont inévitablement soumises à des tensions dangereuses.

**DANGER !**

Haute tension !

Danger de mort !

- Respecter les prescriptions générales d'installation et de sécurité lors des travaux effectués sur des installations de puissance.
- Après l'installation, vérifier le raccordement du conducteur de protection à tous les appareils électriques conformément au schéma de raccordement.
- Aucune exploitation n'est autorisée, même pour des opérations de mesure et de vérification de courte durée, si le conducteur de protection n'est pas raccordé de manière fixe à tous les composants électriques.
- Avant de procéder à des travaux sur des pièces électriques soumises à des tensions supérieures à 50 V, séparer l'appareil du réseau ou de toute autre source de tension. Prendre les mesures nécessaires pour interdire tout réenclenchement. Après la coupure, vérifier la tension à l'aide d'un instrument de mesure. Assurer que la tension est inférieure à 50 V avant d'accéder aux composants.
- Ne pas toucher les points de raccordement électrique des composants à l'état enclenché.
- Avant la mise sous tension de l'appareil, protéger les parties conductrices par des écrans afin d'interdire tout contact dangereux.
- Prévoir une protection contre les contacts indirects (EN 50178, section 5.3.2).

**DANGER !**

Courants de fuite élevés !

Danger de mort !

- Les courants de fuite sont supérieurs à 3,5 mA. Les appareils doivent donc être raccordés de manière fixe au réseau d'alimentation (selon EN 50178 – Equipement des installations de puissance).

PD_SicherhMax_fr.fm

## 2.5.3 Très basses tensions à séparation sûre

*PELV*
***P**rotective-**E**xtra-**L**ow-**V**oltage*

La tension du signal et la tension de commande des systèmes PacDrive™ est de < 33 V. Dans cette plage, la spécification en tant que système PELV selon IEC 364-4-41 comprend une mesure de protection contre les contacts directs et indirects avec des tensions dangereuses. Cette protection est assurée par une « séparation sûre » entre côté primaire et secondaire. ELAU recommandent fortement de prévoir l'installation /la machine avec sécurité de séparation.

**DANGER !**

Tensions électriques élevées résultant des défauts de câblage !

Danger de mort ou blessures graves !

- Veillez à connecter aux branchements de tension de signalisation uniquement des appareils, des composants électriques ou des câbles garantissants une sécurité de séparation suffisante des circuits raccordés selon les normes (EN 50178 / édition 1998 - Equipement électronique utilisé dans les installations de puissance).
- Assurez que la sécurité de séparation présente soit maintenue sur la totalité du circuit.

*FELV*
***F**unctional-**E**xtra-**L**ow-**V**oltage*

Lors de l'usage des composants ELAU dans des systèmes sans sécurité de séparation en tant que protection contre les contacts directs ou indirects avec des tensions dangereuses, un capot de protection permanent est exigé pour tous les branchements et contacts (p.ex. MAx-4, connecteur Sub-D, interface série) qui ne sont pas conformes à la classe de protection IP2X. La conception du capot de protection ou du câblage de l'appareil raccordé doit permettre l'enlèvement du capot ou du câblage uniquement à l'aide d'un outil. Cette mesure de protection doit être respectée sur tous les appareils branchés.

## 2.5.4 Mouvements dangereux

Les causes de mouvements dangereux peuvent être diverses :

- défauts de câblage,
- erreurs dans les programmes d'application,
- défauts des composants,
- défauts des capteurs de mesure et de signaux,
- fausses manœuvres.

La protection des personnes doit être assurée par des dispositifs de surveillance ou des mesures situés à un niveau supérieur de l'installation. Il ne faut en aucun cas se fier aux seules surveillances internes des composants de l'entraînement. Les surveillances ou mesures ne doivent être prévues par le constructeur qu'en fonction des conditions spécifiques à l'installation et sur la base d'une analyse des risques et des défaillances potentielles. Elles doivent bien entendu respecter les prescriptions de sécurité applicables à l'installation.

**DANGER !**

Mouvements dangereux !

Danger de mort, blessures graves ou dommages matériels importants !

- Interdire l'accès aux zones dangereuses au moyen, par exemple, de clôtures de sécurité, de grilles de protection, d'écrans de protection ou de barrières lumineuses.
- Prévoir un dimensionnement suffisant des dispositifs de sécurité.
- Disposer l'interrupteur d'arrêt d'urgence de manière telle qu'il soit facilement et rapidement accessible. Le fonctionnement du dispositif d'arrêt d'urgence doit être vérifié avant la mise en service et lors des contrôles périodiques de maintenance.
- Interdire tout risque de démarrage intempestif par le sectionnement des circuits de puissance des entraînements via un circuit d'arrêt d'urgence ou l'utilisation d'un dispositif fiable de blocage du démarrage.
- Avant d'accéder à la zone dangereuse, veiller à ce que les entraînements soient mis à l'arrêt en toute sécurité.
- Lors des travaux sur l'installation, mettre l'équipement électrique hors tension à l'aide de l'interrupteur général et prendre les mesures nécessaires pour interdire tout réenclenchement.
- L'utilisation d'appareils haute fréquence, de télécommandes ou d'émetteurs/récepteurs radio doit être évitée à proximité de l'électronique des appareils et de ses câbles d'alimentation. Si l'usage de ces appareils est indispensable, il convient de vérifier, avant la première mise en service, qu'ils ne provoquent pas, dans leurs différentes positions, des dysfonctionnements dans le système et l'installation. Si nécessaire, effectuer un contrôle de CEM spécial.

PD_SicherhMax_fr.fm

# 3 Vue d'ensemble du système

## 3.1 Concepts d’entraînement pour les machines d’emballage

Les concepts des machines modernes dans l’industrie de l’emballage sont caractérisés par la demande de dynamique élevée, flexibilité, modularité et rentabilité. Les machines d’emballage étaient équipées traditionnellement d’une motorisation principale et d’un arbre mécanique couplant tous les autres organes de mouvements de la machine avec des fonctions mécaniques compliquées. La construction de telles machines d’emballage flexibles pour des produits différents est très onéreuse. Toutes modifications même insignifiantes liées au processus d’emballage et surtout au changement de produits nécessitent des transformations chères et surtout l’arrêt de la machine.

Les machines d’emballage avec un arbre électronique permettent une flexibilité totale. Des systèmes d’entraînements par servomoteurs avec électroniques remplacent les cames mécaniques et la pignonerie ; un axe électronique virtuel assure le synchronisme des axes indépendants. Tous les déplacements et mouvements en synchronisme cyclique ou angulaire sont contrôlés par une commande centralisée.

Des états des machines non systématiques, tels que par exemple des situations de blocages ou d’arrêt d'urgence et des mouvements d’initialisation, peuvent être réalisés en synchronisme angulaire. Des modifications dynamiques de produits ou des matériaux d'emballage dans la machine, comme par exemple un glissement des produits à emballer ou un allongement du matériel d’emballage, peuvent être surveillées par des capteurs et corrigées par des séquences pendant la marche de la machine. Le concept des machines mécaniques classiques va être transformé et simplifié fortement par ce développement. Les machines d’emballage peut être divisées en éléments standards modulaires.

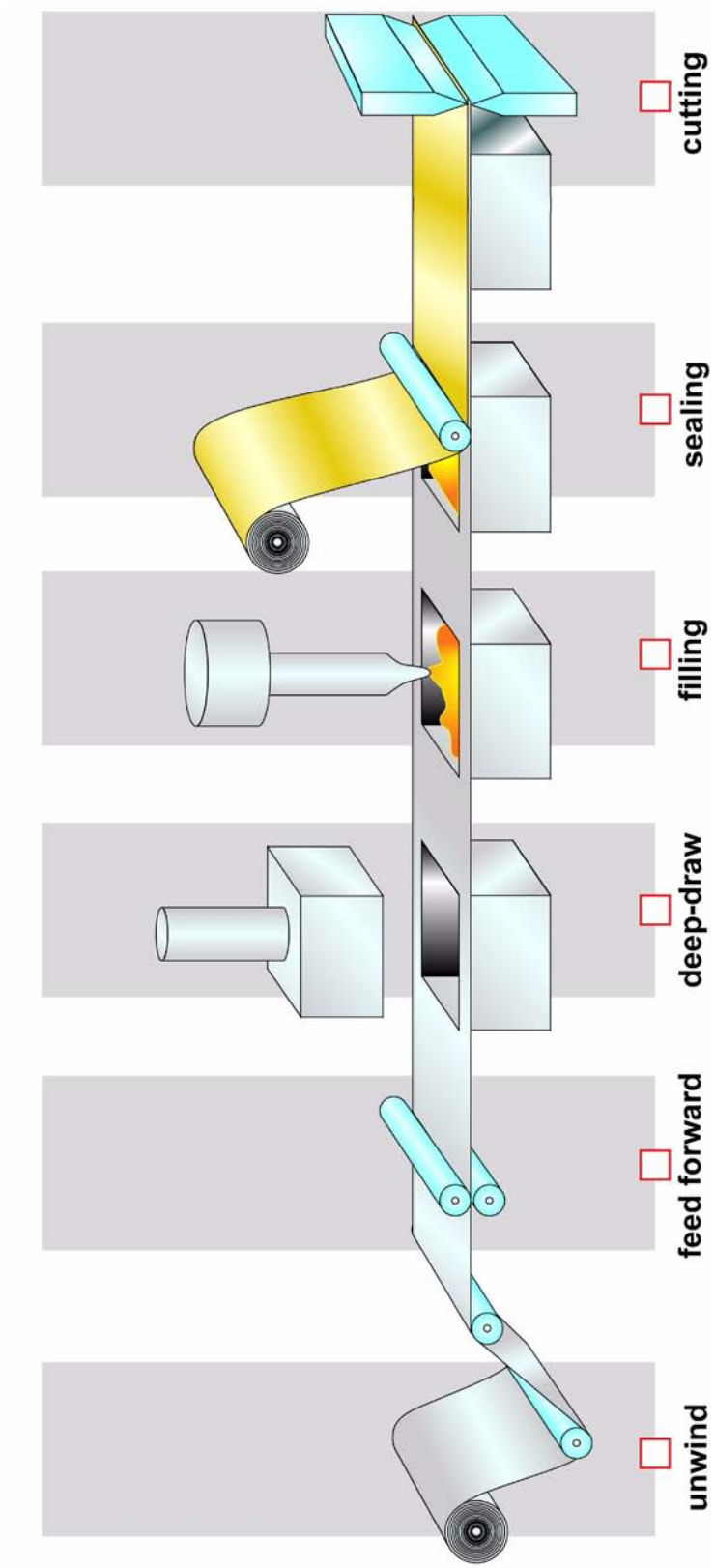



*Illustration 3-1: Dessin de projet d'une machine d'emballage*

PD_SysUebers_fr.fm

## 3.2 Structure du système d’automatisation PacDrive™

Le système d’automatisation PacDrive™ offre une solution optimale technique et économique pour des machines d’emballage électroniques. Le système PacDrive™ est composé d'une commande efficace avec une architecture à base de PC ou dispositif de contrôle, du PacDrive Controller (p.ex. MAx-4) et des drives « MotorController MC-4 » numériques qui comprennent l'alimentation puissance, l'étage final de puissance et la régulation servo des axes indépendants (Illustration 3-2).

Le PacDrive Controller est la tête intelligente du système et basé sur un PC industriel. Le PacDrive Controller synchronise et coordonne les fonctions de mouvements de la machine d'emballage. L’utilisation d'un API logiciel IEC 1131-3 permet de rentrer dans des domaines qui étaient réservés jusqu’ici à un automate classique. Les tâches indépendantes d'automates et de mouvements peuvent être divisées en tâches parallèles différentes qui vont être implémentées par l'environnement de programmation EPAS-4 selon le standard IEC 1131-3. On peut relier jusqu'à 40 axes servo au PacDrive Controller et leur envoyer des ordres de déplacement indépendants.

Le bus de terrain en temps réel numérique et circulaire SERCOS assure la communication avec les drives « MotorController MC-4 ». L'utilisation de fibres optiques pour la communication rend le bus insensible aux parasites électromagnétiques. De manière cyclique, il transmet aux drives « MotorControllers MC-4 » décentralisés des valeurs de consignes à une vitesse de 4 Mbaud. Tous les états internes des axes peuvent être contrôlés par le bus en temps réel et peuvent être assimilés par le PacDrive Controller.

Le PacDrive Controller comprend en plus des entrées / sorties numériques et analogiques deux interfaces séries et une interface Ethernet. L'interface OPC intégrée permet la liaison avec des systèmes des visualisation de procès et des systèmes maître PacDrive™. D'autres groupes périphériques peuvent être reliés par modules d'interface bus de terrain. Un PacDrive Controller peut être maître ou esclave du bus de terrain.

Les bus de terrain standards internationaux CANopen, PROFIBUS-DP et Device-Net sont supportés. Les interfaces installées rendent possible le diagnostic par modem téléphonique ou Internet. Les PC peuvent communiquer avec le PacDrive Controller et diagnostiquer directement les états de commande par le TCP/IP.

## 3.3 Concept

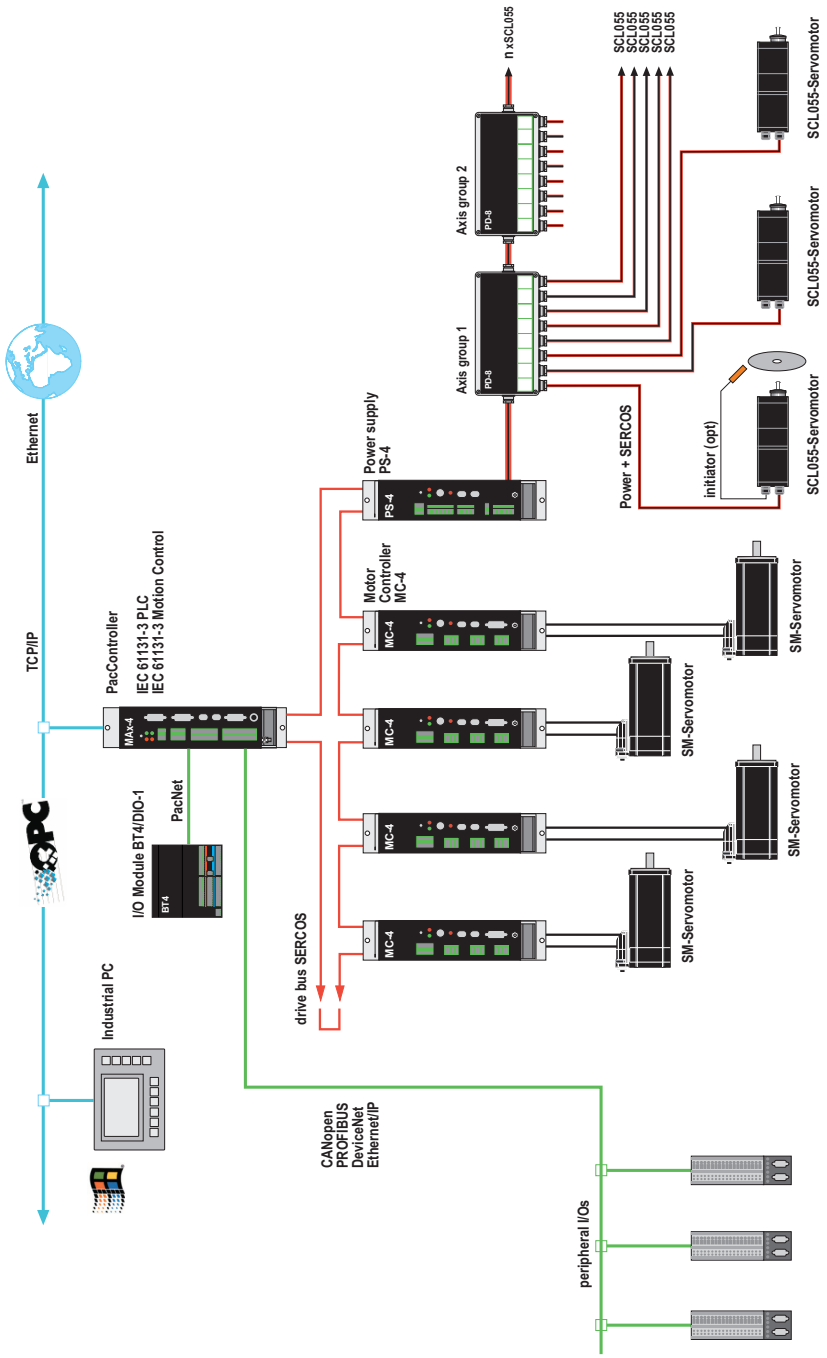



*Illustration 3-2: Vue d'ensemble du système d'automatisation PacDrive™*

Le PacDrive Controller peut être couplé alternativement à l'aide d'un bus de terrain et avec un réseau TCP/IP conventionnel (supérieur au PacDrive Controller).

PD_SysUebers_fr.fm

## 3.4 Composants

### Logiciel de programmation Automation Toolkit (Atelier logiciel) EPAS-4

Le logiciel de programmation EPAS-4 se distingue par un grande variété de fonctions et outils sophistiqués. Un avantage décisif de l'EPAS-4 est la synchronisation interne de tous les composants.

Cela signifie pour vous, l'utilisateur :

Période d'initiation réduite, facilité de manoeuvre, intégration de tous les outils.

***Bibliothèques*** Les bibliothèques complexes accordées au groupe ciblé contribuent à économiser des frais et améliorent la qualité des programmes d'application.

Les atouts du logiciel de programmation Automation Toolkit EPAS-4

- Capable de fonctionner sous Windows (Windows XP recommandé)
- Langages de programmation IEC 61131-3
- Outil SCOPE (fonctions oscilloscope)
- Outil de diagnostic
- Très bonnes possibilités de mise au point et de test
- Liaison série ou TCP / IP avec le PacDrive Controller

### Famille des PacDrive Controller

***Fonctionnalité vaste*** Le PacDrive Controller, un matériel de commande à base de microprocesseur avec le système d'exploitation en temps réel VxWorks, assure de manière centralisée les fonctions d'automate et de mouvement.

Un PacDrive Controller synchronise, coordonne et génère les fonctions de mouvements pour un maximum de 44 entraînements d'une machine d'alimentation et d'emballage.

Pour des tâches HMI on utilise les HMI standards les plus divers. Peu importe s'il s'agit de texte clair économique ou d'IPC, ces tâches ne posent aucun problème à la flexibilité des PacDrive Controller.

En plus, le PacDrive Controller P600 est équipé d'un véritable PC. Grâce à son architecture à base de PC, il peut en outre des fonctions de mouvement usuels accomplir facilement les tâches HMI qui se présentent.

### Drive « MotorController MC-4 »

***Technologie avancée***

Le drive « MotorController MC-4 » numérique se distingue par une exécution compacte et autarcique, à montage mural, ainsi que par une technologie très avancée. Sur le drive MC-4 innovatif, l'alimentation puissance, l'étage final de puissance et la régulation servo logiciel pour un axe sont disposés dans un boîtier peu encombrant. Communiquant avec le PacDrive Controller exclusivement via des fibres optiques, ce drive convient également à une structure décentralisée. Il ne nécessite aucun programme utilisateur, traite des codeurs Singleturn ou Multiturn et subit une autoconfiguration à l'aide de la plaque signalétique sur le moteur SM.

Les atouts du drive « MotorController MC-4 »

- Plage de tension adaptée à une utilisation à l'échelle mondiale
- Bloc d'alimentation intégré
- Puissance max. 34,5 / 69 kVA
- Reconnaissance automatique du moteur
- Taille minimale
- Entrée de sécurité Inverter Enable
- Surcharge 250 %
- Interface SERCOS intégrée
- Nombre de types réduit

### Moteur SM

***Servomoteurs d'une dynamique très élevée***

Des machines à cycle rapide nécessitent des servomoteurs AC d'une dynamique très élevée.
La série de moteurs SM représente pour vous, l'utilisateur, un concept de moteurs optimal pour vos machines d'alimentation et d'emballage. Les servomoteurs dynamiques et sans balais sont pourvus de codeurs de haute résolution (Singleturn ou Multiturn) et d'une plaque signalétique électronique. Une surface lisse et une taille peu encombrante correspondent aux exigences du marché cible.

Les atouts des moteurs SM

- Faible moment d'inertie
- Capacité de surcharge quadruple
- Technique haute tension
- Technique magnétique avancée
- Codeur de haute résolution Singleturn ou Multiturn
- Plaque signalétique électronique
- Boîte à bornes
- Degré de protection IP 65

# 4 Bases de la norme IEC 61131

## 4.1 Automates programmables

Apparus vers le début des années 80, les automates programmables (AP) sont devenus, au fil des années, des éléments centraux de la commande d'automatismes.

Grâce à leurs multiples avantages, tels que rapidité d'adaptation aux changements de processus, facilité d'extension et simplicité de maintenance, ils se sont rapidement imposés face aux commandes à relais classiques.

Le fonctionnement et la structure des systèmes est indépendante de la taille de l'automate programmable. Un automate programmable se compose d'une partie matérielle et d'une partie logicielle. Le matériel comprend un processeur associé à des modules de mémoire et un ensemble de circuits électroniques destinés au raccordement des modules d'entrée et de sortie. Les automates modulaires se composent de différents modules, comme le rack équipé du bus système, le bloc d'alimentation, l'unité centrale avec la mémoire utilisateur, les modules d'entrée/sortie tout-ou-rien et les modules intelligents pour le traitement des valeurs analogiques ou la commande d'entraînements.
Un niveau de tension standardisé de 24 V a été défini pour le niveau d'entrée/sortie. Il a permis d'obtenir une immunité aux perturbations adaptée aux niveaux de parasites susceptibles d'apparaître en milieu industriel.
Le logiciel comprend le système d'exploitation et le programme utilisateur. Le système d'exploitation gère les ressources du système et les fonctions d'organisation. Il assure en outre le démarrage contrôlé du système à la mise sous tension, la gestion des défauts et l'échange des informations via les modules de communication. Il coordonne enfin le traitement du programme utilisateur, qui représente le déroulement logique de la tâche de commande.

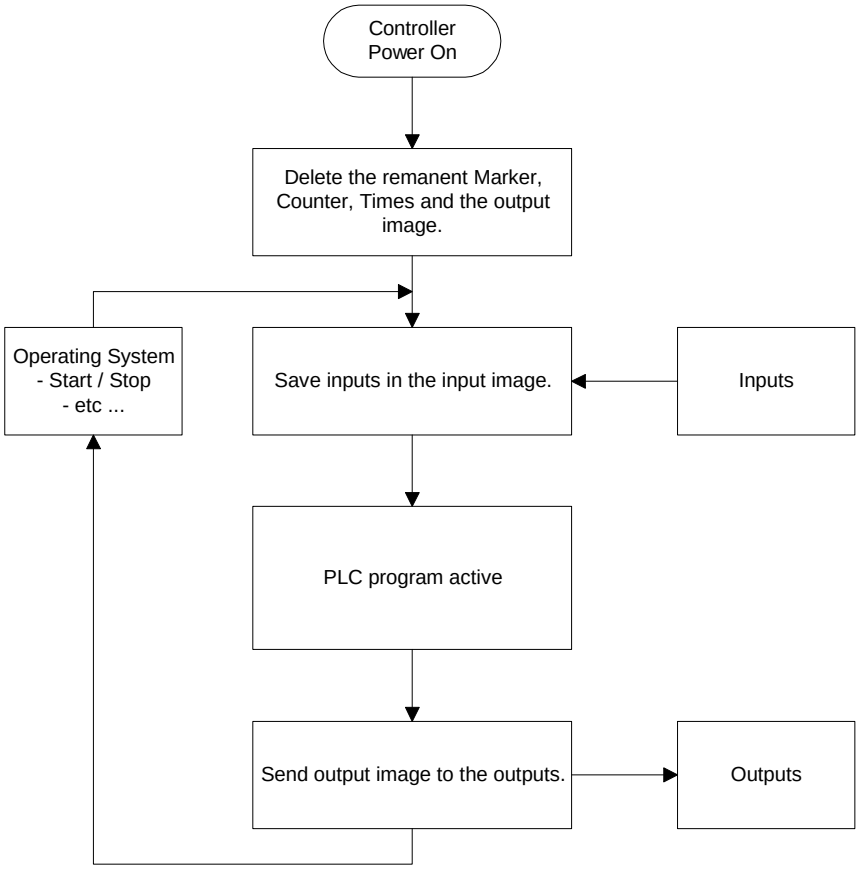



*Illustration 4-1: Cycle de traitement d'un automate programmable classique*

A la mise sous tension, l'automate lance aussitôt le traitement du programme utilisateur (en mode RUN). Toutes les mémoires non rémanentes sont remises à zéro et le cycle commence par la lecture du registre-image d'entrée. Il se poursuit par le traitement de l'ensemble de la liste d'instructions. Au terme du traitement, les résultats déposés dans le registre-image de sortie sont transmis aux sorties et le cycle de traitement du programme recommence par la lecture du registre-image d'entrée. Le temps de cycle représente le principal critère de vitesse d'un automate. Il se situe en pratique entre quelques millisecondes et quelques centaines de millisecondes.

**REMARQUE**

Le cycle de traitement décrit ici s'applique principalement aux systèmes monotâches. Le système PacDrive étant un système multitâches, des divergences peuvent apparaître.
Voir également « Particularités du PacDrive™ » à la page 92.

La simplicité du cycle de traitement constitue un atout majeur, car elle permet de manipuler les données et les programmes en cours de fonctionnement. Presque tous les environnements de programmation admettent la modification de l'ensemble des variables en cours de cycle. La structure fixe du programme rend également possible le chargement de séquences de programme en cours de traitement (*Online Change*), car il existe toujours des points d'atterrissage exacts en début et en fin de cycle.

Lors du lancement sur le marché des automates programmables, les constructeurs se sont demandé comment familiariser leurs clients, habitués de longue date aux commandes à contacteurs, à la programmation des nouveaux systèmes. On a ainsi assisté à de nombreuses tentatives, le plus souvent éphémères, de mise au point d'un langage de programmation permettant de passer facilement du schéma électrique classique au programme d'automate. L'origine des langages de programmation à *blocs fonctionnels*, à *schémas à contacts* et en *listes d'instructions* remonte à cette époque.

Les constructeurs ont manqué alors l'occasion unique de définir un standard de programmation universel applicable à tous les automates programmables. Chacun a continué de cultiver ses particularités dans les différents langages et des standards « constructeur » se sont imposés dans certaines régions, comme Siemens STEP 5 en Europe, Allen Bradley aux Etats-Unis ou MITSUBISHI en Asie.

## 4.2 IEC 61131

Au cours de ces dernières années, la complexité des applications et, par conséquent, le travail de programmation se sont considérablement accrus. Malheureusement, la possibilité de réutiliser des programmes déjà établis était compromise par la multiplicité des standards spécifiques aux constructeurs.
C'est la raison pour laquelle, au début des années 90, les principaux intéressés se sont attachés à mettre au point, sous l'égide de la *Commission Electrotechnique Internationale* (IEC), une norme adoptée sous la référence *IEC 61131*. Celle-ci définit, outre les langages de programmation déjà connus des *schémas à contacts* (LD), *blocs fonctionnels* (FBD) et *listes d'instructions* (IL), les nouveaux langages *littéral structuré* (ST) et *diagramme fonctionnel en séquence* (SFC). La norme IEC 61131 dresse l'inventaire des exigences auxquelles doit répondre un système d'automate programmable moderne. Elle ne doit cependant pas être considérée comme un spécification rigide, mais comme un ligne directrice pour la programmation des automates. C'est dans le même esprit qu'elle décrit les principales caractéristiques d'un automate, tout en laissant aux constructeurs suffisamment de champ libre pour mettre en œuvre leurs propres développements.

La norme IEC 61131 tente de transposer les techniques logicielles modernes au monde des automates programmables. Cette démarche était devenue urgente en raison de la complexité croissante des processus et des fonctions ainsi que de l'explosion des coûts de développement des programmes utilisateur.
La possibilité de réutilisation de modules logiciels testés et standardisés apparaissait comme la seule solution. La mise en œuvre de ce principe était cependant rendue difficile par les caractéristiques suivantes :

- adressage direct
- variables non typées
- absence de vérification de type

Les objectifs de la norme IEC 61131 peuvent se résumer en quelques points :

- Utilisation de techniques logicielles permettant la programmation de modules logiciels réutilisables
- Analyse des problèmes dans leur globalité
- Décomposition des tâches complexes en modules transparents
- Définition d'interfaces claires
- Standardisation des langages dans le but d'augmenter la portabilité

PDM_UserMan_IEC_fr.fm

### 4.2.1 Le modèle de programmation

La norme IEC 61131 fournit, outre des éléments destinés à la programmation et à l'organisation du programme utilisateur, des lignes directrices destinées à la modélisation et à la structuration des systèmes d'automates programmables. Pour la structuration du système, elle fait appel aux notions de *configuration* et de *ressource*.
Le modèle prend également en compte des propriétés telles que des systèmes multiprocesseurs, des systèmes modernes d'automates programmables avec traitement multitâches, un nombre illimité d'entrées et sorties tout-ou-rien et analogiques, une capacité de communication avec d'autres automates programmables ou des ordinateurs.
Une configuration définit la structure d'un appareil. Il peut s'agir, par exemple, d'un automate programmable avec plusieurs unités centrales (UC) éventuellement reliées en réseau, situé au niveau de la cellule machine. Une configuration contient une ou plusieurs ressources correspondant chacune à une commande partielle dotée d'une « fonction de traitement du signal » propre. Dans une configuration réelle, une ressource est représentée par une unité centrale d'automate généralement apte à fonctionner en multitâches.
La structuration d'une ressource s'effectue à l'aide d'un ou plusieurs programmes exécutés sous le contrôle de tâches. On entend par tâche une unité de programme exécutable à laquelle sont associés une priorité et un type d'exécution. Cela permet de formuler, au sein d'un programme, des commandes d'exécution de caractéristiques différentes. On peut ainsi programmer non seulement des tâches cycliques avec un temps de cycle uniforme, mais aussi combiner des temps de cycles ou fournir au système des unités de programme commandées par des événements. La répartition du temps de l'unité centrale à l'intérieur d'une ressource est définie par le niveau de priorité des tâches.
L'affectation de programmes à une tâche a une incidence sur le temps d'exécution du programme général, qui peut être traité indépendamment dans une unité centrale. Grâce à la flexibilité du système modulaire, un programme peut appartenir à plusieurs tâches. Plusieurs instances avec différents temps d'exécution sont alors générées.

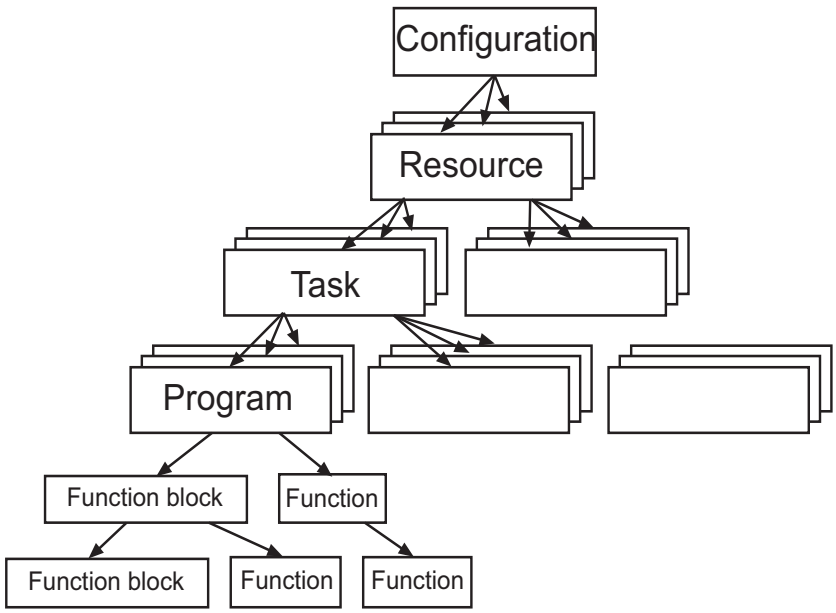



*Illustration 4-2: Le modèle de programmation selon IEC 61131*

La norme IEC 61131-3 définit des données locales, qui peuvent être déclarées dans des programmes, des blocs fonctionnels ou des fonctions. Les données locales ne sont accessibles qu'au sein d'une unité d'organisation du programme et représentent donc un mécanisme d'encapsulation des données. Il existe par ailleurs des données globales, qui sont accessibles à toutes les unités d'organisation de programme à l'intérieur d'une ressource. En cas d'utilisation de systèmes multitâches, l' accès à des données globales risque cependant d'aboutir à des incohérences de données. On dispose en outre de données directement accessibles dotées d'adresses fixes à l'intérieur de la plage d'adressage de l'automate. Il s'agit en général des adresses des entrées (Input, I), sorties (Output, O) et mémoires internes (M).
Un autre aspect du modèle de programmation concerne le comportement au démarrage de l'automate. La norme décrit la *reprise à froid* et la *reprise à chaud*. En cas de reprise à froid, le programme est entièrement rechargé. Toutes les variables sont remises à leur valeur initiale. La valeur initiale peut être une valeur par défaut ou une valeur spécifiée par l'utilisateur. Toutes les tâches de la ressource sont relancées. En cas de reprise à chaud (redémarrage), les variables ne sont pas remises à leur valeur initiale, mais reprennent la valeur qu'elles avaient au moment de l'interruption.

PDM_UserMan_IEC_fr.fm

### 4.2.2 Le modèle de communication

L'échange de données représente un autre aspect important de la description des éléments structurels. Le modèle de communication défini par la norme IEC 61131 permet d'établir des programmes bien structurés et surtout modularisés. Ces propriétés sont fondamentales pour le développement de modules de programme conformes aux besoins de l'utilisateur et réutilisables. La norme IEC 61131 prévoit les possibilités de communication suivantes :

- Chemins d'accès (VAR_ACCESS)
- Variables globales (VAR_GLOBAL, VAR_EXTERNAL)
- Paramètres d'appel
- Blocs fonctionnels de communication (IEC 61131-5)

Tous les éléments d'une configuration communiquent entre eux ou avec d'autres systèmes à l'aide de *chemins d'accès* définis. Des *variables globales* sont en outre utilisées pour faire communiquer simplement des programmes au sein d'une configuration. Les variables globales peuvent être créées et utilisées au niveau de la configuration, de la ressource et du programme.
L'échange de données à l'intérieur de programmes s'effectue à l'aide de *paramètres d'appel*, de variables d'entrée/sortie ou de valeurs de fonctions. Cette méthode bien connue des programmeurs en langages évolués constitue une innovation dans la programmation des automates programmables. Les paramètres d'appel et les variables de transfert permettent de définir des interfaces claires, apportant ainsi une contribution importante à l'encapsulation des fonctionnalités.
Outre ces éléments du modèle de communication, il est également possible d'utiliser des « blocs fonctionnels de communication » spéciaux. Ces blocs de nature monolithique gèrent les échanges de données entre émetteur et récepteur de manière entièrement autonome. Les services de communication sont définis dans la partie 5 de la norme IEC 1131, qui est encore en cours de préparation.
Si l'on considère le modèle de communication de la norme IEC1131, on constate que les blocs logiciels standardisés ont été largement privilégiés. Grâce à l'encapsulation des fonctionnalités et des données, à des interfaces clairement définies et à un comportement sans risque d'interaction avec le reste du programme, les modules sont de mieux en mieux accueillis par les utilisateurs.

La définition de types de données représente l'une des standardisations importantes de la norme IEC 61131-3. La norme reconnaît différents types de données élémentaires, à partir desquels il est possible de spécifier des types de données dérivés ou définis par l'utilisateur.

Lors de la programmation, l'utilisateur peut utiliser des types de données standards et des types de données qu'il a lui-même définis. A chaque identificateur est affecté un type de donnée, qui détermine la place mémoire à réserver et les valeurs à associer au contenu de la mémoire.

### Types de données élémentaires

La norme IEC 1131-3 définit cinq groupes de types de données élémentaires, dont le type générique est indiqué entre parenthèses.

- Cordon de bits (ANY_BIT)
- Entier signé et non signé (ANY_INT)
- Nombre réel (ANY_REAL)
- Date, heure (ANY_DATE)
- Cordon de caractères, durée, dérivé (ANY)

| Type de donnée | Description | Nombre de bits | Etendue | Valeur initiale |
|---|---|---|---|---|
| BOOL | Booléen | 1 | [0, 1] | 0 |
| BYTE | Cordon de 8 bits | 8 | [0, ..., 255] | 0 |
| WORD | Cordon de 16 bits | 16 | [0, ..., 65535] | 0 |
| DWORD | Cordon de 32 bits | 32 | [0, ..., 4,295 E09] | 0 |
| LWORD | Cordon de 64 bits | 64 | [0, ..., 1,845 E19] | 0 |

*Tableau 4-1: Cordons de bits*

| Type de donnée | Description | Nombre de bits | Etendue | Valeur initiale |
|---|---|---|---|---|
| SINT | Entier court | 8 | [-128, ..., +127] | 0 |
| INT | Entier | 16 | [-32768, ..., +32767] | 0 |
| DINT | Entier double | 32 | [$-2^{31}$, ..., $+2^{31}-1$] | 0 |
| LINT | Entier long | 64 | [$-2^{63}$, ..., $+2^{63}-1$] | 0 |
| USINT | Entier court non signé | 8 | [0, ..., +255] | 0 |
| UINT | Entier non signé | 16 | [0, ..., +65535] | 0 |
| UDINT | Entier double non signé | 32 | [0, ..., $+2^{32}-1$] | 0 |
| ULINT | Entier long non signé | 64 | [0, ..., $+2^{64}-1$] | 0 |

*Tableau 4-2: Entiers signés et non signés*

PDM_UserMan_IEC_fr.fm

| Type de donnée | Description | Nombre de bits | Etendue | Valeur initiale |
|---|---|---|---|---|
| REAL | Nombre réel | 32 | voir IEC 559 | 0.0 |
| LREAL | Nombre réel long | 64 | voir IEC 559 | 0.0 |

*Tableau 4-3: Tableau : Nombres réels*

| Type de donnée | Description | Nombre de bits | Etendue | Valeur initiale |
|---|---|---|---|---|
| DATE | Date | comme DWORD | | 0001-01-01 |
| TOD | Heure du jour | comme DWORD | | 00:00:00 |
| DT | Date et heure du jour | comme DWORD | | 0001-01-01-00:00:00 |

*Tableau 4-4: Date, heure*

| Type de donnée | Description | Nombre de bits | Etendue | Valeur initiale |
|---|---|---|---|---|
| TIME | Durée | comme DWORD | | 0 s |
| STRING | Cordon de caractères | par défaut = 80 | | cordon « vide » |

*Tableau 4-5: Durée, cordon de caractères*

Exemple de déclaration d'un cordon de caractères :
str:STRING(35):='Ceci est un cordon'



**REMARQUE**

Les types de données LWORD, LINT et ULINT ne sont pas gérés par le système PacDrive™ .

### Notation des types de données

Les types de données mentionnés ci-dessus peuvent être représentés de différentes façons :

BOOL, BYTE, WORD, DWORD, LWORD

Ces types de données peuvent être représentés comme suit :

- TRUE ou 1
- FALSE ou 0
- Représentation décimale, hexadécimale (16#), octale (8#) oder binaire (2#)

Exemple pour WORD : 234, 16#ff, 2#1001_1100_0011_1111

SINT, INT, DINT, LINT, USINT, UINT, UDINT, ULINT

Représentation décimale, hexadécimale (16#), octale (8#) ou binaire (2#). Le trait de soulignement (_) sépare les unités.

Exemples :

Représentation décimale pour INT : -123, +234, 0, 1_000
Représentation hexadécimale pour INT : 16#F1, 16#0A_1B
Représentation binaire pour INT : 2#0001_0011_0111_1111

REAL, LREAL

Représentation décimale normale avec point décimal ou représentation exponentielle.

Exemple : 1000.23 et 1.23e3 et 1.23E3 et 1.23E03 sont interprétés de manière identique.

TIME (durée)

TIME#, t# ou T# se trouve au début d'une désignation de date/heure. Le dépassement est autorisé (par ex. 25 heures).
d indique les jours, h les heures, m les minutes, s les secondes et ms les millisecondes. Le trait de soulignement (_) sépare les unités.

Exemple : T#2d_26h_4m_12s_123ms

DATE, TIME_OF_DAY ou TOD, DATE_AND_TIME ou DT

DATE# ou D# indique une date.
TIME_OF_DAY# ou TOD# indique une heure du jour.
DATE_AND_TIME# ou DT# indique une date et une heure du jour.
Date : D#1998-12-07 signifie 7 juillet 1998.
Notation heure du jour : TOD#12:00:00.123
Date et heure : 1998-12-07-12:00:00.123

STRING

Des apostrophes ' ' encadrent le cordon de caractères.
Le signe dollar $ introduit des caractères de commande (changement de ligne, tabulation…).

Exemples :
Cordon de caractères, caractère de commande : 'This is a line feed character $L'
Cordon vide : ''

PDM_UserMan_IEC_fr.fm

**Types de données dérivés**

Selon la norme IEC 1131-3, des types de données peuvent être dérivés des types de données élémentaires. Ces types de données dérivés sont les suivants :

- Tableaux (ARRAY)
- Pointeur (POINTER)
- Types de données énumérés
- Structures (STRUCT)
- Références

Les types de données dérivés permettent de gérer des structures complexes à l'aide de constructions simples : des données complexes peuvent ainsi être transmises à des fonctions et blocs fonctionnels à l'aide d'une seule affectation. Les types de données dérivés peuvent être créés à partir des types de données de base ou de types de données dérivés.

Tableaux (ARRAY)

Un tableau se compose d'un ensemble de variables individuelles appartenant au même type de donnée.
Le système admet les tableaux de types de données élémentaires à une, deux et trois dimensions. Les tableaux peuvent être définis dans la partie déclaration d'un bloc ou dans la liste de variables globales.

Syntaxe :

<Nom_Tableau>: **ARRAY** [<li1>..<ls1>,<li2>..<ls2>] **OF** <Type. elem>.

li1, li2 indiquent les limites inférieures de l'étendue des valeurs d'indices du tableau, ls1, ls2 les limites supérieures. Les valeurs limites doivent être des nombres entiers.

Exemple :

JeuCartes : **ARRAY** [1..13, 1..4] **OF** INT;

Initialisation de tableaux :
Tous les éléments d'un tableau sont initialisés en même temps.

Exemples d'initialisation de tableaux :

```
arr1 : ARRAY [1..5] OF INT := 1,2,3,4,5;
arr2 : ARRAY [1..2,3..4] OF INT := 1,3(7);
               (* représentation abrégée de 1,7,7,7 *)
arr3 : ARRAY [1..2,2..3,3..4] OF INT := 2(0),4(4),2,3;
               (* représentation abrégée de
  0,0,4,4,4,4,2,3 *)
```

L'accès aux éléments d'un tableau bidimensionnel s'effectue à l'aide de la syntaxe suivante :

<Nom_Tableau>[Indice1,Indice2]

Exemple :

```
JeuCartes [9,2]
```

PDM_UserMan_IEC_fr.fm

Pointeur (POINTER)



**REMARQUE**

L'utilisation de pointeurs est déconseillée, car elle ne fait l'objet d'aucune vérification lors de la compilation et en cours de traitement. La transmission de paramètres par VAR_IN_OUT constitue généralement la meilleure solution.

Les pointeurs possèdent un type de donnée et une adresse. Le type de donnée correspond au type de donnée de l'élément de donnée vers lequel pointe le pointeur. L'adresse correspond à l'adresse (l'emplacement) à laquelle l'élément de donnée est mémorisé.

On mémorise dans les pointeurs des adresses de variables ou de blocs fonctionnels au moment du traitement du programme.

La déclaration des pointeurs s'effectue selon la syntaxe :

<Identificateur>: **POINTER TO** <TypeDonnee/BlocFonctionnel>;

Un pointeur peut pointer vers n'importe quel type de donnée et de bloc fonctionnel, même définis par l'utilisateur.

L'opérateur ADR permet d'affecter au pointeur l'adresse d'une variable ou d'un bloc fonctionnel.

Le déréférencement d'un pointeur s'effectue à l'aide de l'opérateur « ^ » placé après l'identificateur du pointeur.

Exemple :

```
pt:POINTER TO INT;
var_int1:INT := 5;
var_int2:INT;
pt := ADR(var_int1);
var_int2:= pt^; (* var_int2 est maintenant 5 *)
```

Type de donnée énuméré

Un type de donnée énuméré est un type de donnée défini par l'utilisateur, qui se compose d'un ensemble de constantes de type cordon de caractères. Ces constantes sont appelées valeurs d'énumération.

Les valeurs d'énumération sont connues dans l'ensemble du projet, même si elles ont été déclarées localement dans un bloc. L'énumération commence par le mot clé TYPE et se termine par END_TYPE.

Syntaxe :

**TYPE** <Identificateur>:(<Enum_0> ,<Enum_1>, ...,<Enum_n>);

**END_TYPE**

L'<identificateur> peut prendre l'une des valeurs d'énumération et est initialisé avec la première valeur. Ces valeurs étant compatibles avec les entiers, elles permettent d'effectuer les mêmes opérations

qu'avec INT. Un nombre x peut être affecté à l'<identificateur>. Si les valeurs d'énumération ne sont pas initialisées, le comptage commence à 0. Veillez, lors de l'initialisation, à ce que les valeurs initiales soit croissantes. La validité des nombres est vérifiée en cours de traitement.

Exemple :

```
FEU: (Rouge, Orange, Vert:=10); (*Rouge a la valeur
  initiale 0, Orange 1, Vert 10 *)
FEU:=0; (* Le feu a la valeur Rouge*)
FOR i:= Rouge TO Vert DO
  i := i + 1;
END_FOR;
```

La même valeur d'énumération ne doit pas être utilisée deux fois.

Exemple :

```
FEU: (Rouge, Orange, Vert);
COULEUR: (Bleu, Blanc, Rouge);
```

Erreur : Rouge ne doit pas être utilisé à la fois pour FEU et COULEUR.

Structures (STRUCT)

Une structure se compose de plusieurs éléments de données individuels qui constituent un ensemble du point de vue de l'utilisateur. Les différents éléments peuvent appartenir à des types de données différents.

Les structures commencent par le mot clé TYPE et se terminent par END_TYPE.

Les déclarations de structures obéissent à la syntaxe suivante :

**TYPE** <NomStructure>:

**STRUCT**

<DeclarationVariables 1>

.

.

<DeclarationVariables n>

**END_STRUCT**

**END_TYPE**

Les structures imbriquées sont autorisées. La seule limitation réside dans le fait que les variables ne sont pas associables à des adresses (la déclaration AT n'est pas autorisée).

Exemple de définition d'une structure nommée TracagePolygone :

```
TYPE TracagePolygone:
STRUCT
  Depart:ARRAY [1..2] OF INT;
  Point1:ARRAY [1..2] OF INT;
```

```
   Point2:ARRAY [1..2] OF INT;
   Point3:ARRAY [1..2] OF INT;
   Point4:ARRAY [1..2] OF INT;
   Fin:ARRAY [1..2] OF INT;
END_STRUCT
END_TYPE
```

L'accès aux éléments d'une structure s'effectue à l'aide de la syntaxe suivante :

<NomStructure>.<NomElement>

Si l'on a, par exemple, une structure nommée « Semaine » contenant un élément nommé « Lundi », il est possible d'accéder comme suit à cet élément :

Semaine.Lundi

Références

Le type de donnée Référence, qui est défini par l'utilisateur, sert à affecter un autre nom à une variable, une constante ou un bloc fonctionnel.

Les références commencent par le mot clé TYPE et se terminent par END_TYPE.

Syntaxe :

**TYPE** <Identificateur>: <ExpressionAffectation>;

**END_TYPE**

Exemple :

```
TYPE message:STRING[50];
END_TYPE;
```

### Déclaration de variables et de constantes

Les variables sont des identificateurs définis par l'utilisateur. Elles servent à désigner des données dans le programme de l'automate. Elles se caractérisent essentiellement par leur contenu variable. Conformément à la norme IEC 61131-3, les variables s'utilisent pour la mémorisation et le traitement des informations.
La norme IEC 61131-3 distingue cinq classes de variables :

- Variables globales,
- Variables locales,
- Variables d'entrée,
- Variables de sortie,
- Variables d'entrée et sortie.

Les variables d'entrée, de sortie et d'entrée/sortie se rapportent à un programme, une fonction ou un bloc fonctionnel. Elles ne peuvent être utilisées que dans le mode spécifiquement défini. A l'intérieur de l'unité de programme à laquelle elles sont associées, elles sont accessibles en lecture et en écriture ; à l'extérieur, elles ne sont accessibles que dans le mode défini.

La norme IEC 61131-3 définit un identificateur comme un cordon de lettres, de chiffres et de caractères de soulignement (_). Ce cordon doit commencer par une lettre ou un caractère de soulignement. Les identificateurs ne doivent pas comporter d'espaces, de caractères spéciaux ou de caractères accentués. Ils peuvent se composer de 32 caractères significatifs. Aucune distinction n'est faite entre majuscules et minuscules.

Les variables définies par l'utilisateur doivent être déclarées. Toutes les variables utilisées dans une UOP doivent être spécifiées dans la partie déclaration de l'UOP.

La partie déclaration doit être introduite, dans la forme littérale, par l'un des mots clés suivants :

- VAR
- VAR_INPUT
- VAR_OUTPUT

.Les déclarations doivent être séparées par des points-virgules et se terminer par le mot clé VAR_END.

| Mot clé | Utilisation des variables |
|---|---|
| VAR | Interne à l'unité d'organisation |
| VAR_INPUT | Fournie de l'extérieur ; non modifiable dans l'unité d'organisation |
| VAR_OUTPUT | Fournie par l'unité d'organisation aux entités externes |
| VAR_IN_OUT | Fournie par des entités externes ; peut être modifiée dans l'unité d'organisation |
| VAR_EXTERNAL | Fournie par la configuration ; peut être modifiée dans l'unité d'organisation |
| VAR_GLOBAL | Déclaration de variables globales |

*Tableau 4-6: Mots clés de déclaration de variables*

### Attributs de variables

| Mot clé | Utilisation des variables |
|---|---|
| RETAIN | Variable non volatile (protégée contre les coupures de tension) |
| CONSTANT | Constante (variable qui ne peut être modifiée) |
| AT | Affectation d'un emplacement de mémoire |

*Tableau 4-7: Mots clés de déclaration de variables / d'attributs*

| Priorité | Affectation | Comportement au démarrage |
|---|---|---|
| 1 (la plus élevée) | Valeur retenue pour les variables non volatiles (attribut RETAIN) | *Reprise à chaud* : Restauration de la valeur au retour de la tension ou après un arrêt |
| 2 | Valeur initiale spécifiée par déclaration | *Reprise à froid* : Valeur initiale pour redémarrage défini |
| 3 (la plus faible) | Valeur initiale associée au type de donnée (par défaut) | *Reprise à froid* : Valeur initiale pour redémarrage défini |

*Tableau 4-8: Affectation des valeurs initiales selon niveau de priorité*



**REMARQUE**

De nouvelles valeurs initiales peuvent être définies pour les types de données dérivés.

### Les variables possèdent différents champs d'application (domaines de validité)

Les variables peuvent être déclarées en dehors d'une UOP et être utilisées dans tout le programme, figurer dans une UOP en tant que paramètre d'appel ou être accessibles localement à l'intérieur d'une UOP.
Lorsqu'une variable est déclarée dans un bloc fonctionnel, son champ d'application est limité à ce bloc fonctionnel. Lorsqu'elle est, en revanche, déclarée dans un programme, elle est accessible à tous les blocs fonctionnels qui sont déclarés dans ce programme.

Variables
<Nom>:<TYPE>;
Exemple : Test: BOOL;

Constantes
VAR CONSTANT <Nom>:<TYPE>:=<Valeur>
Exemple : VAR CONSTANT Test: BOOL:=TRUE;

### Variables à adressage fixe (variables directement représentées)

Ces variables s'appuient sur la technique classique des automates. Elles sont relativement importantes dans les systèmes IEC 61131-3, car :

- tous les signaux du processus sont reliés par des variables à adressage fixe,
- les chevauchements de variables à adressage fixe sont autorisés et peuvent être utilisés comme moyen de programmation.

PDM_UserMan_IEC_fr.fm

L’affectation d’un emplacement de mémoire fixe à une variable est réalisée à l’aide du mot clé « AT ». Elle définit trois paramètres :

- l’adresse de début (après le qualificatif « AT », en tant que décalage à partir de 0),
- la plage de mémoire (entrée, sortie, mémoire interne),
- la longueur de la variable dans la mémoire (à l’aide d’une abréviation de type).

Dans cet énoncé,

- le premier caractère de la définition est un « % »,
- le deuxième est un préfixe
  - I pour entrée (Input), Q pour sortie (Output), M pour mémoire interne,
  - longueur de la variable dans la mémoire (à l’aide d’une abréviation de type de variable),
- le troisième définit la taille :
  - X pour un seul bit (l’adresse du bit N se présente toujours sous la forme « Octet.N », par exemple 1.0),
  - B pour octet,
  - W pour mot,
  - D pour double-mot et
  - L pour mot long.

Exemples :

```
%IB24, %QX1.1, %MW12
```

### Chevauchement de variables

Le chevauchement de variables est autorisé : %MB12 correspond, par exemple, au premier octet de %MW12 et au premier octet de %MD12. Ce chevauchement est également utilisable au niveau du bit : %MX12.0 est, par exemple, le premier bit (de plus faible poids) de %MB12.

### Initialisation de variables

Toutes les variables sont initialisées lors d’une reprise à froid. La valeur par défaut est généralement 0 ou FALSE. L’utilisateur peut bien sûr spécifier une autre valeur initiale en utilisant l’opérateur d’affectation « := » dans la déclaration.
L’initialisation peut également se faire à l’intérieur de types de données dérivés (tableaux, structures). L’exemple suivant montre la syntaxe à utiliser.

Exemple :

```
VAR
a : INT := 13;
b : STRING := ‘this is a string‘;
c : REAL := 1.1;
```

```
END_VAR

VAR
a : myStruct :=
   (
   status := TRUE,
   inputValue := 2.5
   );
END_VAR

VAR
a : ARRAY[1..10] OF INT :=
   1, 2, 2(4), 5, 6, 7, 8, 9,10;
END_VAR
```

## 4.2.3 Unités d'organisation de programme UOP

La norme IEC 61131-3 a réduit la diversité des types de modules utilisés dans les modèles de programmation spécifiques aux constructeurs. La définition d'unités d'organisation de programme (UOP) a considérablement simplifié et homogénéisé l'organisation du programme.
Dans les systèmes utilisés jusqu'ici, la récupération de modules pour la programmation d'automates d'autres constructeurs était une tâche quasi-impossible, même pour les programmeurs les plus expérimentés, en raison de la signification implicite et non transparente des différents modules et plages.
En structurant l'UOP en *programme*, *fonction* et *bloc fonctionnel*, la norme a limité le nombre de niveaux afin d'augmenter la lisibilité des programmes et faciliter la gestion globale de l'application.

### Programme

Ce type d'UOP représente le « programme principal ». Toutes les variables du programme auxquelles sont affectées des adresses physiques (entrées et sorties de l'AP, par exemple) doivent être déclarées dans cette UOP ou dans le niveau de structure supérieur (*ressource* ou *configuration*). Accès aux variables globales et aux chemins d'accès.

### Fonction

Une *fonction* décrit une logique de traitement complexe, mais qui est dépourvue de « mémoire », c'est-à-dire de variables statiques. Une *fonction* se caractérise par le fait qu'elle fournit toujours le même résultat avec des paramètres d'entrée identiques.

Exemple :

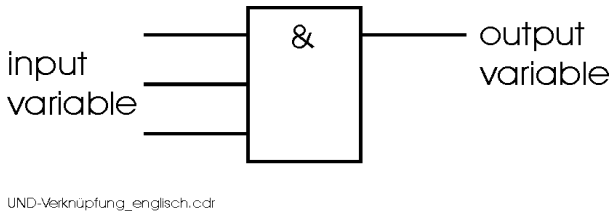



*Illustration 4-3: L'opération logique ET est une fonction booléenne standard*

### Bloc fonctionnel

Si l'on a besoin d'un module doté de mémoire, le *bloc fonctionnel* (FB) offre la base nécessaire grâce à ses variables statiques locales. Un FB (compteur ou temporisateur, par exemple) peut fournir des résultats différents avec des paramètres d'entrée identiques. Chaque instance d'un FB possède, selon la nouvelle norme, sa propre plage de données protégée, sur laquelle s'effectuent les calculs : l'instance (voir ci-dessous).
Pour standardiser les fonctionnalités typiques des automates, la

norme a défini un ensemble de fonctions et de blocs fonctionnels standards. Cette *bibliothèque* constitue une base importante qui ouvre les portes à une programmation harmonisée et indépendante de tout constructeur.

Exemple :
La temporisation d'enclenchement TON (Timer ON) est un bloc fonctionnel standard. Lorsque la valeur booléenne « 1 » est appliquée à l'entrée INPUT (IN), la temporisation définie à l'entrée PT est lancée. Après écoulement de la temporisation, la sortie Q prend la valeur booléenne « 1 ». La valeur instantanée (temps écoulé) peut être lue à la sortie ET.

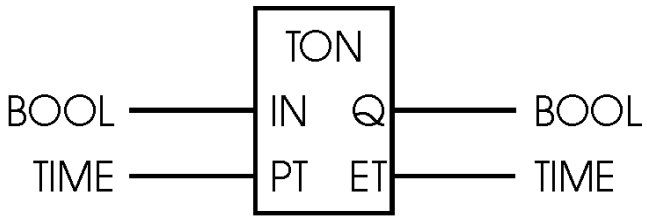



*Illustration 4-4: Temporisation d'enclenchement TON (Timer ON)*

En résumé, on peut considérer une UOP comme une unité qui peut être convertie par le compilateur indépendamment d'autres parties de programmes. Les caractéristiques de l'UOP permettent d'élaborer des programmes utilisateur très modularisés et de récupérer des modules logiciels déjà implémentés et testés. Pour permettre à des modules de programme d'accéder à des UOP, il est nécessaire de déclarer au minimum l'interface d'appel (prototype). Les parties de programme compilées peuvent ensuite être reliées logiquement pour constituer un programme cohérent (liens). Contrairement à de nombreux langages évolués, la norme IEC 61131 ne reconnaît pas de domaine de validité aux UOP. Le nom d'une UOP est global dans un projet et ne peut pas être affecté plusieurs fois. Une fois déclarée, une UOP est accessible globalement à toutes les autres UOP.

**Structure d'une unité d'organisation de programme**

Une UOP se compose d'une partie déclaration et d'une partie instructions.
La partie déclaration contient la déclaration des variables.
Elle est suivie de la partie instructions, qui énumère les commandes qui doivent être exécutées.

| Mot clé | UOP | Commentaire |
|---|---|---|
| PROGRAM | Programme | Il s'agit d'un programme principal permettant l'accès à la périphérie de l'automate. Les variables globales et les chemins d'accès sont déclarés ici. |
| FUNCTION_BLOCK | Bloc fonctionnel | Possède des variables d'entrée et de sortie ; s'utilise pour la programmation de tâches fréquemment utilisées. |
| FUNCTION | Fonction | Module logiciel simple permettant d'étendre le jeu d'opérations de l'automate. |
| END_FUNCTION | | Marque de fin d'une UOP (fonction, dans ce cas). |

*Tableau 4-9: Les unités d'organisation de programme et leurs mots clés*

Fonctions

Les fonctions peuvent être codées dans tous les langages définis par la norme IEC 61131-3, à l'exception du diagramme fonctionnel en séquence (SFC). La valeur à renvoyer est déclarée après le nom de la fonction.

Exemple :

```
FUNCTION TEST: REAL
(* -> Le type de donnée de la valeur à renvoyer est
   REAL *)
(* Partie déclaration *)
VAR_INPUT
   Bouton1: BOOL;
   Bouton2: BOOL;
END_VAR
VAR
   ResultInterm: BOOL;
END_VAR
(* Partie instructions *)
TEST:= Bouton1 AND Bouton2;
(* -> Affecter la valeur renvoyée par la fonction *)
END_FUNCTION
```

**REMARQUE**

La norme IEC 61131-3 définit des fonctions standards qui sont utilisées par la majorité des automates. Ces fonctions standards sont décrites de manière détaillée dans l'aide en ligne EPAS-4.

Blocs fonctionnels

Les blocs fonctionnels s'utilisent pour positionner des entrées, des sorties et des mémoires internes ; les états d'un appel de bloc fonctionnel sont mémorisés de cycle en cycle. Le code programme d'un bloc fonctionnel entraîne ainsi des modifications des entrées et des sorties ainsi que des mémoires internes.
Le programme appelant n'a accès qu'aux variables d'entrée et de sortie du bloc fonctionnel. Les appels d'autres blocs fonctionnels sont autorisés, et ce, quel que soit le langage.

Exemple :

```
FUNCTION_BLOCK Counter
(* Partie déclaration *)
VAR_INPUT
   iMode : INT; (* 0 = Reset, 1 = Count *)
END_VAR
VAR_OUTPUT
   iCounter : INT; (*Valeur réelle de comptage *)
END_VAR
(* Partie instructions *)
IF iMode = 0 THEN
   iCounter:= 0; (*Reset*)
ELSEIF Mode = 1 THEN
   iCounter:= iCounter + 1;
ENDIF;
END_FUNCTION_BLOCK
```



**REMARQUE**

La norme IEC 61131-3 définit des blocs fonctionnels standards qui sont utilisés par la majorité des automates. Ces blocs fonctionnels standards sont décrits de manière détaillée dans l'aide en ligne EPAS-4.

La norme IEC 61131-3 prévoit l'instanciation des blocs fonctionnels (FB). Une instance est une structure dans laquelle toutes les mémoires internes, entrées et sorties d'un appel de FB sont mémorisées. Ainsi, un programme appelant cinq fois FB1 possédera 5 instances de FB1 (une pour chaque appel). L'avantage de ce procédé *orienté objet* inhabituel réside dans le fait que le diagnostic du programme peut s'effectuer en fonction d'un appel précis et sans effets secondaires.Les outils modernes aident le programmeur à effectuer cette instanciation grâce à une déclaration automatique : à chaque appel d'un FB, le système crée un nom d'instance qui gère les données de cet appel.
Il est important de noter que toutes les instances utilisent le même code programme du FB. Les modifications effectuées sur le code programme se répercutent par conséquent de la même manière sur tous les appels. Une instance est donc une copie du FB destinée à un appel.

PDM_UserMan_IEC_fr.fm

Exemple :

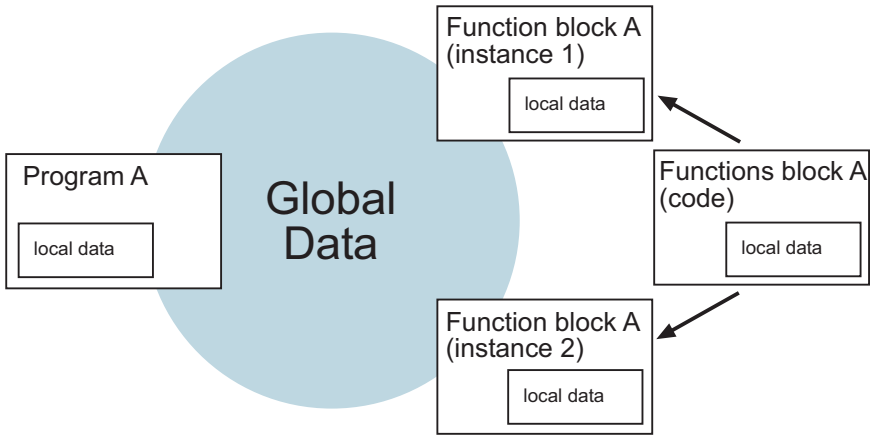



*Illustration 4-5: Instanciation*

Une copie de la plage de données est réalisée pour chaque instance.

Le bloc fonctionnel A est instancé dans deux appels.Il existe pour chaque appel une structure (portant le nom de l'instance) qui contient les valeurs des variables spécifiques à l'appel.

Dans l'exemple ci-dessus, l'instance 1 et l'instance 2 sont des données locales (structures) dans le programme appelant. Ces données d'instance locales peuvent servir de variables d'entrée à d'autres blocs fonctionnels ou programmes.
Méthode nouvelle pour le programmeur d'automate traditionnel, l'instanciation permet un traitement des variables sans effets annexes et un diagnostic à l'aide d'outils de programmation modernes basés sur PC. Le confort du diagnostic justifie le travail supplémentaire exigé par l'instanciation. Dans les environnements d'automates anciens, il fallait copier plusieurs fois le code d'un FB, puis procéder aux modifications du programme dans chacune des copies du code.

Programme

Les programmes sont les unités d'organisation de programme de hiérarchie supérieure. Ils appellent les fonctions, les blocs fonctionnels ou même d'autres programmes dans certaines applications. Ils peuvent être écrits dans tous les langages.
Contrairement aux blocs fonctionnels, les programmes ne sont pas instancés. Ils ne possèdent aucune fonction de mémorisation des données locales pour le cas où ils seraient appelés plusieurs fois.

Exemple de programme :

```
PROGRAM Main
(* Partie déclaration *)
```

```
VAR
  counter_1 : Counter; (* Instance du FB Counter *)
  iActCount : INT;
END_VAR
(* Partie instructions *)
IF bfirstCycle THEN
  counter_1(Mode := 0);
  (* -> Appel du FB Counter avec mode Reset *)
ELSE
  counter_1(Mode := 1);
  (* -> Appel du FB Counter avec mode comptage *)
END_IF
actCount := counter_1.Out;
  (* -> Accès à la variable de sortie de counter_1*)
END_PROGRAM
```

## 4.2.4 Les langages de programmation

Les langages de programmation utilisés sont le diagramme fonctionnel en séquence (SFC), la liste d'instructions (IL), le schéma à contacts (LD), le langage en blocs fonctionnels (FBD) et langage littéral structuré (ST). Chaque langage a ses domaines d'application particuliers ou s'avère plus adapté à la résolution de certains problèmes. La norme définit la syntaxe des langages de programmation avec tant d'exactitude que l'utilisateur peut partir du principe que tous les logiciels de développement conformes à IEC 61131 utilisent au moins une syntaxe identique.

**Liste d'instructions (IL)**

La liste d'instructions IL (Instruction List) fait un peu figure d'assembleur parmi les langages de programmation d'automates. Ancêtre de la programmation des automates, elle est essentiellement répandue dans les pays européens. Elle convient plus particulièrement au traitement des programmes séquentiels simples. Dès que des constructions en boucle sont nécessaires, la liste d'instructions tend à perdre de sa lisibilité.

La liste d'instructions est un langage de bas niveau qui s'appuie sur des anciens assembleurs et travaille avec des accumulateurs. Une seule action (chargement dans l'accumulateur, mémorisation dans le registre) peut être effectuée par ligne de code programme. Le contrôle des flux, comme les branchements, est réalisé à l'aide de sauts conditionnels et inconditionnels et d'étiquettes de saut. Des commentaires peuvent être ajoutés sur les lignes après les instructions.

Chaque instruction débute sur une nouvelle ligne et doit contenir un opérateur et, selon le type d'opération, un ou plusieurs opérandes séparés par des virgules.
L'instruction peut être précédée d'une étiquette d'identification suivie de deux points (:).
Si un commentaire est présent, il doit constituer le dernier élément d'une ligne. Des lignes vides peuvent être insérées entre les instructions.

Exemple :

| Etiquette de saut | Opérateur | Opérande | Commentaire |
|---|---|---|---|
| | LD | 17 | |
| | ST | lint | (* Commentaire *) |
| | GE | 5 | |
| | JMPC | next | |
| | LD | idword | |
| | EQ | instruct.sdword | |
| | STN | test | |
| next: | | | |

Modificateurs et opérateurs en IL

Les opérateurs et modificateurs suivants peuvent être utilisés en langage IL.

Modificateurs :

- C avec JMP, CAL, RET : l'instruction n'est exécutée que si le résultat de l'expression précédente est TRUE.
- N avec JMPC, CALC, RETC : l'instruction n'est exécutée que si le résultat de l'expression précédente est FALSE.
- N dans les autres cas : négation de l'opérande (et non de l'accumulateur)

Le tableau suivant énumère tous les opérateurs autorisés en IL ainsi que leurs modificateurs possibles :

| Opérateur | Modificateurs | Signification |
|---|---|---|
| LD | N | Rendre le résultat courant égal à l'opérande |
| ST | N | Mémoriser le résultat courant à l'emplacement de l'opérande |
| S | | Positionner l'opérande booléen à TRUE si le résultat courant est TRUE |
| R | | Remettre l'opérande booléen à FALSE si le résultat courant est TRUE |
| AND | N, ( | ET booléen |
| OR | N, ( | OU booléen |
| XOR | N, ( | OU exclusif booléen |
| ADD | ( | Addition |
| SUB | ( | Soustraction |
| MUL | ( | Multiplication |
| DIV | ( | Division |
| GT | ( | > |
| GE | ( | >= |
| EQ | ( | = |
| NE | ( | <> |
| LE | ( | <= |
| LT | ( | < |
| JMP | C, N | Saut vers l'étiquette |
| CAL | C, N | Appel d'un bloc fonctionnel |
| RET | C, N | Retour d'un bloc fonctionnel appelé |
| ) | | Evaluation d'une opération différée |

*Tableau 4-10: Opérateurs en IL*

PDM_UserMan_IEC_fr.fm

Voir également « Opérateurs » à la page 71.

Exemple d'utilisation de modificateurs dans un programme IL :

| Etiquette de saut | Opérateur | Opérande | Commentaire |
|---|---|---|---|
| | LD | TRUE | (* Charger TRUE dans l'accumulateur *) |
| | ANDN | BOOL1 | (* Effectuer l'opération AND avec la valeur inversée de la variable BOOL1 *) |
| | JMPC | etiquette | (* Si le résultat est TRUE, sauter vers l'étiquette « etiquette » *) |
| | LDN | BOOL2 | (* Mémoriser la valeur inversée de *) |
| | ST | RESULT | (* BOOL2 dans RESULT *) |
| etiquette: | LD | BOOL2 | (* Mémoriser la valeur de *) |
| | ST | RESULT | (* BOOL2 dans RESULT *) |

Il est également possible, en IL, de placer des parenthèses après une opération. La valeur de la parenthèse est alors considérée comme opérande.

Exemple :

```
LD   2
MUL  2
ADD  3
ST   Res
```

Res a ici la valeur 7. Si l'on insère des parenthèses :

```
LD   2
MUL( 2
ADD  3
)    ST   Res
```

Res a la valeur 10, car l'opération MUL n'est effectuée que lorsqu'une « ) » est rencontrée ; l'opérande calculé pour MUL est alors 5.

**Littéral structuré (ST)**

Le langage de programmation ST (Littéral structuré) peut être considéré comme un langage évolué comparable au C ou au Pascal. La syntaxe du langage s'appuie sur celle du Pascal. Le langage ST permet une programmation de boucles performante, même sans instructions de saut, et l'insertion efficace de fonctions mathématiques.Il comporte aussi bien des itérations (For, While, Repeat) que des instructions conditionnelles (If ... Then ... Else, Case). L'expérience pratique montre que les constructions ST – analogues à celles du Pascal – sont particulièrement lisibles et compréhensibles. C'est la raison pour laquelle les programmeurs européens optent très souvent pour ce langage lorsqu'ils peuvent choisir entre la diversité de langages offerts par la norme IEC1131.

Le langage ST (littéral structuré) se compose d'un série d'instructions qui, comme dans les langages évolués, peuvent être conditionnelles (« IF..THEN..ELSE ») ou itératives (WHILE..DO).

Les instructions se terminent toujours par un point-virgule.

Exemple :

```
IF value < 7 THEN
  WHILE value < 8 DO
    value := value + 1;
  END_WHILE;
END_IF;
```

Expressions

Une expression est une construction syntaxique qui fournit une valeur lorsqu'elle est évaluée.

Les expressions se composent d'opérateurs et d'opérandes. L'opérande peut être une constante, une variable, un appel de bloc fonctionnel ou une autre expression.

Evaluation des expressions

L'évaluation d'une expression consiste à traiter les opérateurs selon des règles de priorité définies. L'opérateur ayant la priorité la plus élevée est traité en premier ; il doit être suivi par l'opérateur ayant la priorité immédiatement inférieure et ainsi de suite, jusqu'à ce que tous les opérateurs aient été traités.

Les opérateurs bénéficiant d'une même priorité sont traités de gauche à droite.

Le tableau ci-dessous énumère les opérateurs du langage ST dans leur ordre de priorité.

| Opération | Symbole | Priorité |
|---|---|---|
| Mise entre parenthèses | (<Expression>) | Maximale |
| Appel de fonction | <Nom de la fonction> (<Liste de paramètres>) | |
| Exponentiation | ** | |
| Négation | - | |
| Complémentation | NOT | |
| Multiplication | * | |
| Division | / | |
| Modulo | MOD | |
| Addition | + | |
| Soustraction | - | |
| Comparaison | <,>,<=, >= | |
| Egalité | = | |
| Inégalité | <> | |
| ET booléen | AND | |
| OU exclusif booléen | XOR | |
| OU booléen | OR | Minimale |

*Tableau 4-11: Priorité des opérations*

Les instructions du langage ST, accompagnées d'exemples, sont énumérées dans le tableau suivant :

| Type d'instruction | Exemple |
|---|---|
| Affectation | A:=B; CV := CV + 1; C:=SIN(X); |
| Appel d'un bloc fonctionnel et utilisation de sortie FB | CMD_TMR(IN := %IX5, PT := 300);<br>A:=CMD_TMR.Q |
| RETURN | RETURN; |
| IF | D:=B*B;<br>IF D<0.0 THEN<br>  C:=A;<br>ELSIF D=0.0 THEN<br>  C:=B;<br>ELSE<br>  C:=D;<br>END_IF; |

| Type d'instruction | Exemple |
|---|---|
| CASE | CASE INT1 OF<br>1:BOOL1 := TRUE;<br>2:BOOL2 := TRUE;<br>ELSE<br>BOOL1 := FALSE;<br>BOOL2 := FALSE;<br>END_CASE; |
| FOR | J:=101;<br>FOR I:=1 TO 100 BY 2 DO<br>IF ARR[I] = 70 THEN<br>J:=I;<br>EXIT;<br>END_IF;<br>END_FOR; |
| WHILE | J:=1;<br>WHILE J<= 100 AND ARR[J] <> 70 DO<br>J:=J+2;<br>END_WHILE; |
| REPEAT | J:=-1;<br>REPEAT<br>J:=J+2;<br>UNTIL J= 101 OR ARR[J] = 70<br>END_REPEAT; |
| EXIT | EXIT; |
| Instruction vide | ; |

*Tableau 4-12: Instructions en ST*

Instructions en langage ST

Comme son nom l'indique, le langage littéral structuré (ST) est conçu pour la programmation structurée. Il offre de ce fait des structures préétablies qui facilitent la programmation de constructions fréquemment utilisées, comme les boucles (itérations).

Ce langage présente le double avantage de réduire les probabilités d'erreurs et d'augmenter la lisibilité des programmes.

Comparons, par exemple, deux séquences de programme équivalentes écrites en IL et en ST :

Boucle de calcul en IL :

boucle:

```
LD   Compteur
EQ   0
JMPC fin

LD   Var1
MUL  2
ST   Var1
```

PDM_UserMan_IEC_fr.fm

```
LD   Compteur
SUB  1
ST   Compteur
JMP  boucle

fin:
LD   Var1
ST   Res
```

La même boucle programmée en ST se présenterait comme suit :

```
WHILE Compteur<>0 DO
  Var1:=Var1*2;
  Compteur:=Compteur-1;
END_WHILE

Res:=Var1;
```

On constate que la boucle en ST est non seulement plus courte à programmer, mais aussi beaucoup plus facile à lire. Cette qualité est particulièrement appréciable lorsqu'on a affaire à d'importantes constructions itératives imbriquées.

Les différentes instructions sont décrites ci-après.

Affectation

A gauche de l'opérateur d'affectation « := » se trouve l'opérande (variable, adresse) auquel doit être affectée la valeur de l'expression située à droite.

Exemple :

```
Var1 := Var2 * 10;
```

Après exécution de cette ligne, Var1 a la valeur de Var2 multipliée par 10.

Appel de blocs fonctionnels en ST

L'appel d'un bloc fonctionnel en ST doit s'effectuer à l'aide d'un énoncé comportant le nom de l'instance du bloc fonctionnel suivi d'une liste, entre parenthèses, d'affectations de valeurs aux paramètres d'entrée. L'exemple suivant montre un appel de temporisateur avec affectation de valeurs aux paramètres IN et PT. La variable de résultat Q est ensuite affectée à la variable A.

La variable de résultat est adressée, comme en IL, à l'aide du nom du bloc fonctionnel suivi d'un point et du nom de la variable :

```
CMD_TMR(IN := %IX5, PT := 300);
A:=CMD_TMR.Q
```

Instruction RETURN

L'instruction RETURN s'utilise pour sortir d'une fonction (par exemple, suite à l'évaluation d'une condition).

Exemple :

```
IF COMPTEUR > 10 THEN
  RETURN;
END_IF;
```

Instruction IF ou branchement alternatif

L'instruction IF permet d'exécuter des instructions si une condition est satisfaite.

Syntaxe :

```
IF <Expression_booleenne1> THEN
  <Instructions_IF>
{ELSIF <Expression_booleenne2> THEN
  <Instructions_ELSIF 1>
.
.
ELSIF <Expression_booleenne n> THEN
  <Instructions_ELSIF n-1>
ELSE
  <Instructions_ELSE>}
END_IF;
```

La partie entre accolades {} est optionnelle.

Si l'<Expression_booleenne1> est vraie (TRUE), seules les <Instructions_IF> sont exécutées, et non les autres.

Si la condition n'est pas vérifiée, les expressions booléennes sont évaluées dans l'ordre en commençant par <Expression_booleenne2> jusqu'à ce qu'une expression vraie (TRUE) soit rencontrée. Les instructions placées après cette expression booléenne et avant le prochain ELSE ou ELSIF sont alors traitées.

Si aucune des expressions booléennes n'est vraie (TRUE), seules les <Instructions_ELSE> sont exécutées.

Exemple :

```
IF temp<17 THEN
  chauffage_marche := TRUE;
ELSE
  chauffage_marche := FALSE;
END_IF;
```

Dans ce cas, le chauffage est allumé si la température descend en dessous de 17 degrés, sinon il reste éteint.

Instruction CASE ou sélection multiple

L'instruction CASE permet de rassembler dans une construction plusieurs instructions conditionnelles soumises à une même variable de condition.

Syntaxe :

PDM_UserMan_IEC_fr.fm

```
CASE <Var1> OF
<Valeur 1>:<Instruction 1>
<Valeur 2>:<Instruction 2>
     ...
<Valeur n>:<Instruction n>
ELSE<Instruction ELSE>
END_CASE;
```

Le traitement d'une instruction CASE s'effectue selon le schéma suivant :

- Si la variable contenue dans <Var1> a la <Valeur i>, l'<Instruction i> est exécutée.
- Si <Var 1> n'a aucune des valeurs indiquées, l'<Instruction ELSE> est exécutée.
- Si la même instruction doit être exécutée avec plusieurs valeurs de la variable, ces valeurs peuvent être écrites les unes à la suite des autres séparées par une virgule pour conditionner l'instruction commune.

Exemple :

```
CASE INT1 OF
1, 5:BOOL1 := TRUE;
   BOOL3 := FALSE;
2: BOOL2 := FALSE;
   BOOL3 := TRUE;
ELSE
   BOOL1 := NOT BOOL1;
   BOOL2 := BOOL1 OR BOOL2;
END_CASE;
```

Instruction d'itération FOR ou boucle de comptage

L'instruction d'itération FOR permet de programmer des tâches répétitives.

Syntaxe :

```
Var_INT :INT;
FOR <Var_INT> := <VAL_INITIALE> TO <VAL_FINALE> {BY
   <Largeur de pas>} DO
  <Instructions>
END_FOR;
```

La partie entre accolades {} est facultative.

Les <Instructions> sont exécutées tant que le compteur <Var_INT> ne dépasse pas la valeur <VAL_FINALE>. Cet état est vérifié avant chaque exécution des <Instructions>, de sorte que les <Instructions> ne sont jamais exécutées si <VAL_INITIALE> est supérieur à <VAL_FINALE>.

Après chaque traitement des <Instructions>, <INT_Var> est augmenté de la <Largeur de pas>. La largeur de pas peut avoir

n'importe quelle valeur entière. En l'absence d'indication, celle-ci est positionnée à 1. La boucle se terminera, car la variable <INT_Var> ne peut qu'augmenter.

Exemple :

```
FOR Compteur:=1 TO 5 BY 1 DO
Var1:=Var1*2;
END_FOR;
Res:=Var1;
```

Supposons que la variable Var1 ait été positionnée à la valeur 1. Au terme de la boucle FOR, elle aura la valeur 32.

Instruction WHILE

L'instruction WHILE s'utilise comme l'instruction FOR, mais sa condition d'interruption peut être une expression booléenne quelconque. Lorsque la condition indiquée est vraie, l'exécution de la boucle est lancée.

Syntaxe :

```
WHILE <Expression_booleenne> DO
  <Instructions>
END_WHILE;
```

Les <Instructions> sont exécutées de manière répétitive tant que l'<Expression_booleenne> associée est vraie. Si l'<Expression_booleenne> est fausse dès la première évaluation, les <Instructions> ne seront jamais exécutées. Si l'<Expression_booleenne> ne prend jamais la valeur FALSE, les <Instructions> sont répétées sans fin, ce qui provoque une erreur de temps de traitement.



**REMARQUE**

Le programmeur doit veiller à ce qu'aucune boucle sans fin n'apparaisse en provoquant une modification de la condition dans la partie instructions de la boucle (par une incrémentation ou une décrémentation de compteur, par exemple). En présence d'une boucle sans fin, le PacDrive Controller affiche le message de diagnostic *Dépassement de temps de cycle*.

Exemple :

```
WHILE Compteur<>0 DO
  Var1 := Var1*2;
  Compteur := Compteur-1;
END_WHILE
```

Les instructions d'itération WHILE et REPEAT sont, en un certain sens, plus puissantes que l'instruction FOR, car elles n'imposent pas de connaître le nombre de traitements de la boucle avant son exécution. Dans de nombreux cas, on pourra se limiter à travailler avec ces deux types d'instructions. Si, toutefois, le nombre de traitements de la

boucle est clairement défini, il est préférable d'utiliser l'instruction FOR, car elle n'autorise pas les boucles sans fin.

Instruction REPEAT

La boucle REPEAT se distingue de la boucle WHILE par le fait que la condition d'interruption n'est vérifiée qu'après l'exécution de la boucle. La boucle sera donc traitée au moins une fois, quelle que soit la condition d'interruption.

Syntaxe :

```
REPEAT
  <Instructions>
UNTIL <Expression_booleenne>
END_REPEAT;
```

Les <Instructions> sont exécutées tant que l'<Expression_booleenne> est vraie (TRUE).

Si l'<Expression_booleenne> est vraie dès la première évaluation, les <Instructions> sont exécutées exactement une fois. Si l'<Expression_booleenne> ne prend jamais la valeur TRUE, les <Instructions> sont répétées sans fin, ce qui provoque une erreur de temps de traitement.



**REMARQUE**

Le programmeur doit veiller à ce qu'aucune boucle sans fin n'apparaisse en provoquant une modification de la condition dans la partie instructions de la boucle (par une incrémentation ou une décrémentation de compteur, par exemple). En présence d'une boucle sans fin, le PacDrive Controller affiche le message de diagnostic *Dépassement de temps de cycle*.

Exemple :

```
REPEAT
  Var1 := Var1*2;
  Compteur := Compteur-1;
UNTIL
  Compteur=0
END_REPEAT
```

Instruction EXIT

Lorsque l'instruction EXIT se trouve dans une boucle FOR, WHILE ou REPEAT, la sortie s'effectue à partir de la boucle la plus proche, indépendamment de la condition d'interruption.

**Diagramme fonctionnel en séquence (SFC)**

Le diagramme fonctionnel en séquence SFC (de l'anglais « Sequential Function Chart » ) est un langage graphique destiné à représenter des fonctions de commande séquentielle reliées par des transitions, qui conditionnent le traitement d'actions. La représentation graphique des transitions et des actions rappelle le

schéma fonctionnel, est très lisible et convient particulièrement à la programmation de processus globaux. Contrairement à une idée répandue parmi les programmeurs, le diagramme ne peut pas être entièrement traité au cours d'un cycle de programme. Les calculs complexes et rapides ne peuvent en effet pas être traités correctement en SFC ; ils ne peuvent être exécutés que dans les blocs d'action écrits dans un langage de programmation quelconque.
Les transitions entre ensembles d'actions (étapes) deviennent actives si l'équation booléenne associée est vraie (TRUE). Le langage SFC autorise les branchements sous forme alternative et parallèle :

- les actions dans les branchements alternatifs sont exécutées si la condition d'entrée associée est remplie. L'ordre d'évaluation (priorité) est fixé de gauche à droite.
- les actions dans les branchements parallèles sont toutes lancées simultanément. Une transition à la fin du branchement parallèle définit l'événement de sortie.

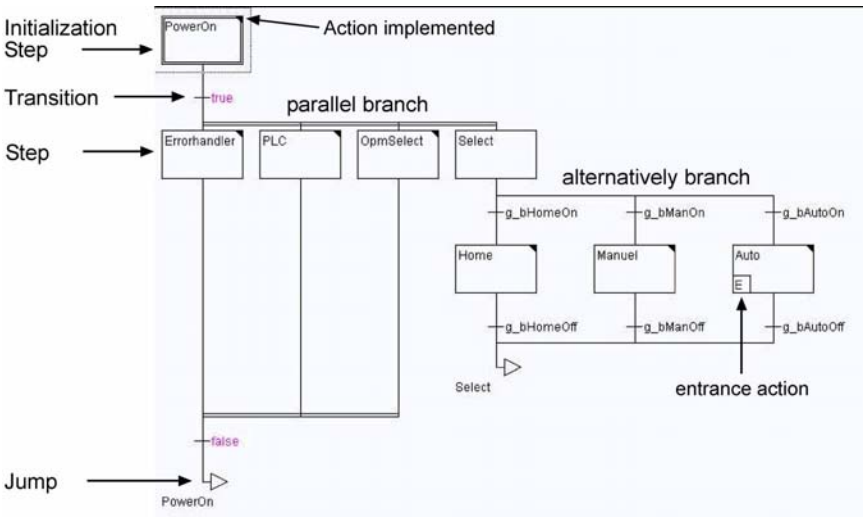



*Illustration 4-6: Exemple de réseau en SFC*

Etape

Un module écrit en langage SFC se compose d'une suite d'étapes reliées entre elles par des liaisons dirigées (transitions).

Il existe deux sortes d'étapes.

- La forme simplifiée se compose d'une action et d'un drapeau qui indique si l'étape est à l'état actif ou inactif. Si l'action associée à une étape est implémentée, un petit triangle apparaît dans le coin supérieur droit de l'étape.
- Une étape IEC se compose d'un drapeau et d'une ou plusieurs actions associées. Les actions associées apparaissent à droite de l'étape. Nous y reviendrons ultérieurement en détail.

Action

Une action peut être une suite d'instructions en langage IL ou ST, un ensemble de réseaux en langage FBD ou LD ou encore un diagramme fonctionnel en séquence (SFC).

Dans les étapes simplifiées, une action est toujours liée à son étape. Pour éditer une action, placez-vous sur l'étape à laquelle l'action est associée, double-cliquez avec la souris ou marquez l'étape et choisissez la commande du menu EXTRAS | ZOOM ACTION / TRANSITION.

Les actions des étapes IEC sont placées directement sous leur bloc SFC dans l'Object Organizer et peuvent être chargées dans leur éditeur par un double clic ou par appui sur la <touche Entrée>. Les nouvelles actions peuvent être créées par PROJECT | ADD ACTION.

Action d'entrée ou de sortie

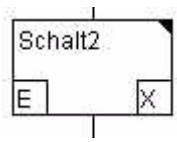
Une action d'entrée et une action de sortie peuvent être ajoutées à une étape. L'action d'entrée n'est exécutée qu'une seule fois, immédiatement après l'activation de l'étape. L'action de sortie n'est exécutée qu'une seule fois avant la désactivation de l'étape.

Une action d'entrée est repérée par un 'E' situé dans le coin inférieur gauche du symbole de l'étape, une action de sortie par un 'X' situé dans le coin inférieur droit.

Les actions d'entrée et de sortie peuvent être programmées dans n'importe quel langage. Pour éditer une action d'entrée ou de sortie, double-cliquez sur le coin correspondant de l'étape.

Les actions d'entrée et de sortie peuvent être définies dans une étape simplifiée, mais non dans une étape IEC.

Exemple d'étape avec action d'entrée et de sortie :




Transitions / Conditions de transition

Les étapes sont reliées par des transitions.

Une condition de transition peut être une variable booléenne, une adresse, une constante ou une suite d'instructions donnant un résultat booléen écrite dans un langage quelconque.

Etape active

Après appel d'un bloc SFC, l'action associée à l'étape initiale (entourée d'une ligne double) est exécutée en premier. Une étape dont l'action est en cours d'exécution est dite active. Lorsqu'une étape est active, l'action qui lui est associée est exécutée une fois par cycle. En mode Online, les étapes actives sont représentées en bleu.

A chaque étape correspond un drapeau chargé de mémoriser l'état de l'étape. Le drapeau d'étape (état actif ou inactif d'une étape) peut être représenté par la valeur logique d'un élément de structure booléen <NomEtape>.x. Cette variable booléenne a la valeur TRUE lorsque l'étape correspondante est active et FALSE lorsqu'elle est inactive. Elle est implicitement déclarée et peut être utilisée dans chaque action et transition du bloc SFC.

Au cours d'un cycle de traitement, toutes les actions appartenant aux étapes actives sont exécutées. Dans un deuxième temps, les différentes étapes suivant les étapes actives deviennent actives si les conditions de transition sont vraies (TRUE).. Les étapes alors actives ne sont traitées qu'au cours du prochain cycle.

Etape IEC

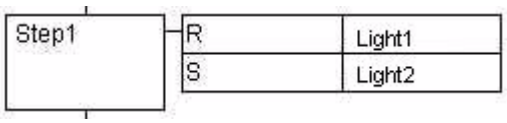
A côté des étapes simplifiées, on dispose dans le langage AS d'étapes conformes à la norme IEC.

Un nombre quelconque d'actions peut être affecté à une étape IEC. Les actions des étapes IEC sont indépendantes des étapes et peuvent être utilisées plusieurs fois au sein des blocs. Elles doivent pour ce faire être associées aux différentes étapes à l'aide de la commande EXTRAS | ASSOCIATE ACTION.

Des variables booléennes peuvent également être affectées aux étapes. Grâce à des qualificatifs, les actions et variables booléennes peuvent être activées et désactivées, avec des temporisations éventuelles. Une action pouvant continuer à être active même lorsque l'étape suivante est en cours de traitement (grâce, par exemple, au qualificatif S), il est possible de programmer des opérations auxiliaires.

Les actions associées à une étape IEC sont représentées à droite de l'étape dans un rectangle divisé en deux parties. La partie de gauche contient le qualificatif, éventuellement accompagné d'une constante de temps, et la partie de droite le nom de l'action.

Exemple d'étape IEC associée à deux actions :




Pour faciliter le suivi des processus, toutes les actions actives sont représentées en bleu en mode online, de la même manière que les étapes actives. Au terme de chaque cycle, le système vérifie quelles sont les actions actives.

Une nouvelle étape insérée est reconnue comme une étape IEC si la commande EXTRAS | USE IEC STEPS est choisie.

Dans l'Object Organizer, les actions sont placées directement en dessous du bloc SFC correspondant. De nouvelles actions peuvent être créées par PROJECT | ADD ACTION.

PDM_UserMan_IEC_fr.fm

**REMARQUE**

Pour pouvoir utiliser les étapes IEC, vous devez intégrer dans votre projet la bibliothèque spéciale SFC *Iecsfc.lib*.

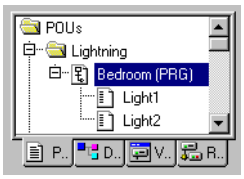



*Illustration 4-7: Bloc SFC avec ses actions dans l'Object Organizer*

Qualificatif

Pour associer des actions à des étapes IEC, on dispose des qualificatifs suivants :

| Qualificatif | Signification | Remarque |
|---|---|---|
| N | Non-stored (non mémorisé) | L'action est active tant que l'étape est active. |
| R | overriding Reset (remise à zéro prioritaire) | L'action est désactivée. |
| S | Set (Stored) (positionné (mémorisé)) | L'action est activée et reste active jusqu'à une remise à zéro. |
| L | time Limited (limité dans le temps) | L'action est activée pour un certain temps. |
| D | time Delayed (temporisé) | L'action est activée après un certain temps dans la mesure où l'étape est encore active. |
| P | Pulse (impulsion) | L'action est exécutée une seule fois lorsque l'étape est activée. |
| SD | Stored and time Delayed (mémorisé et temporisé) | L'action est activée au bout d'un certain temps et reste active jusqu'à une remise à zéro. |
| DS | Delayed and Stored (temporisé et mémorisé) | L'action est activée au bout d'un certain temps dans la mesure où l'étape est encore active et reste active jusqu'à une remise à zéro. |
| SL | Stored and time Limited (mémorisé et limité dans le temps) | L'action est activée pendant un certain temps. |

*Tableau 4-13: Qualificatifs en langage SFC*

Branchement alternatif

Deux ou plusieurs branches en langage SFC peuvent être définies comme branches alternatives. Chaque branche alternative doit commencer et se terminer par une transition. Les branchements alternatifs peuvent contenir des branchements parallèles et d'autres branchements alternatifs. Un branchement alternatif commence par une ligne horizontale (début de l'alternative) et se termine par une ligne horizontale (fin de l'alternative) ou par un saut.

Lorsque l'étape qui précède la ligne de début de l'alternative devient active, la première transition de chaque branche alternative est évaluée de gauche à droite. La première transition de gauche dont la condition de transition prend la valeur TRUE est alors ouverte et l'étape suivante est activée (voir étape active).

Branchement parallèle

Deux ou plusieurs branches en langage SFC peuvent être définies comme branches parallèles. Chaque branche parallèle doit commencer et se terminer par une étape. Les branchements parallèles peuvent contenir des branchements alternatifs ou d'autres branchements parallèles. Un branchement parallèle commence par une double ligne (début du branchement parallèle) et se termine par une double ligne (fin du branchement parallèle) ou par un saut.

Lorsque l'étape qui précède la ligne de début du branchement parallèle devient active et que la condition de transition suivant cette étape prend la valeur TRUE, les premières étapes de toutes les branches parallèles sont activées (voir étape active). L'ensemble de ces branches est alors traité en parallèle. L'étape suivant la ligne de fin du branchement parallèle devient active lorsque toutes les étapes précédentes sont actives et que la condition de transition précédant cette étape prend la valeur TRUE.

Saut

Un saut est une liaison avec une étape dont le nom est indiqué sous le symbole du saut. Les sauts sont nécessaires, car les liaisons ascendantes ou croisées sont interdites.

PDM_UserMan_IEC_fr.fm

Le diagramme suivant montre une évolution simple d'actions en mode mémorisé, normal et Reset :

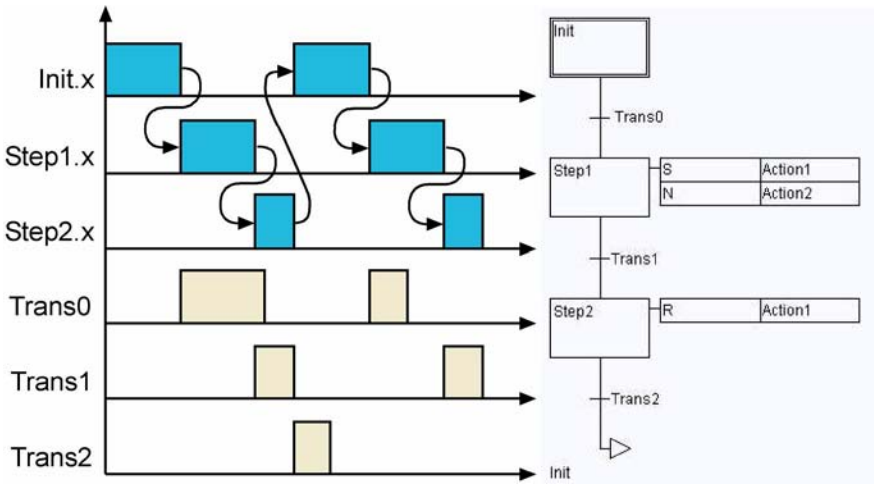



*Illustration 4-8: Les transitions a, b et c commandent l'évolution des étapes 1 et 2.*

Les qualificatifs d'actions commandent l'exécution du code indépendamment de l'étape active : les actions mémorisées ne sont désactivées que dans les étapes suivantes.

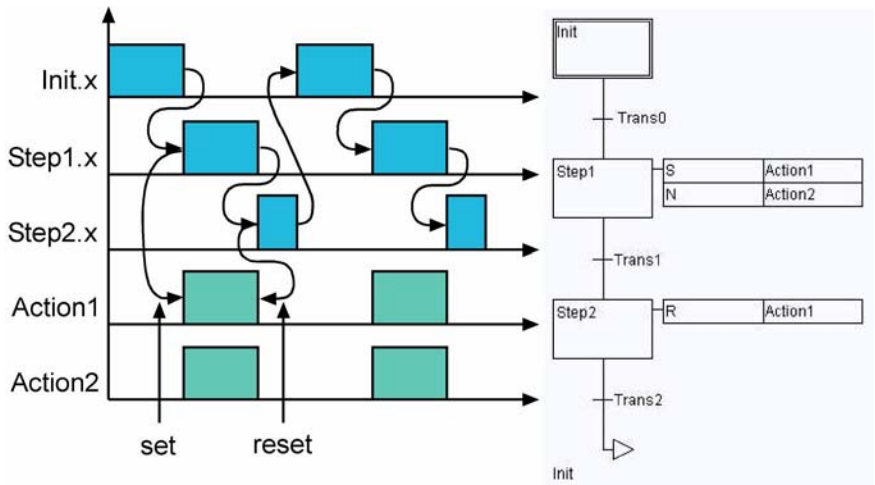



*Illustration 4-9: La fin de l'action 1 est uniquement commandée par le Reset de l'étape 2 et non par la fin de l'étape 1.*

Fonctionnement des qualificatifs L et D : traitement limité dans le temps par L et lancement temporisé du traitement jusqu'à la fin de l'étape par D.

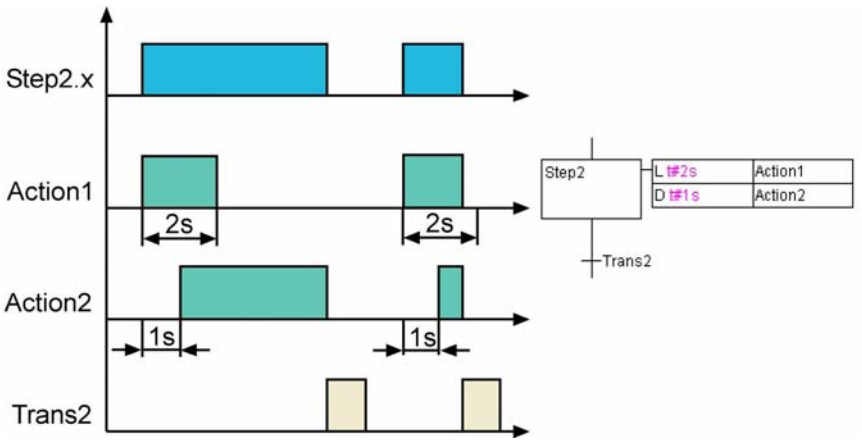



*Illustration 4-10: Qualificatifs d'action L et D pour la commande temporelle d'actions*

### Langage en blocs fonctionnels (FBD)

Le langage en blocs fonctionnels (FBD, de l'anglais « Function Block Diagram ») est un langage graphique qui autorise la programmation de tâches de commande même complexes. Basé sur des blocs fonctionnels définis, il permet de réaliser n'importe quel processus à l'aide d'éléments de liaison. Le flux des données à travers le programme peut être représenté schématiquement, ce qui contribue à accroître la transparence des processus du programme. Les constructions en boucle ou les branchements peuvent cependant poser des problèmes. Des composants matériels étant souvent proposés avec les blocs fonctionnels, on dispose de modules correspondants aux niveaux matériel et logiciel.

Les sorties des blocs fonctionnels sont reliées aux entrées des blocs suivants. Le flux des données peut être représenté sous forme entièrement graphique. Les sauts et les retours facilitent la programmation.

Le langage en blocs fonctionnels est un langage de programmation orienté graphique. Il utilise une liste de réseaux contenant chacun une structure, qui représente une expression logique ou arithmétique, un appel de bloc fonctionnel, un saut ou une instruction Return.

PDM_UserMan_IEC_fr.fm

Exemple de réseau dans le langage en blocs fonctionnels tel qu'il se présenterait dans l'EPAS-4 :

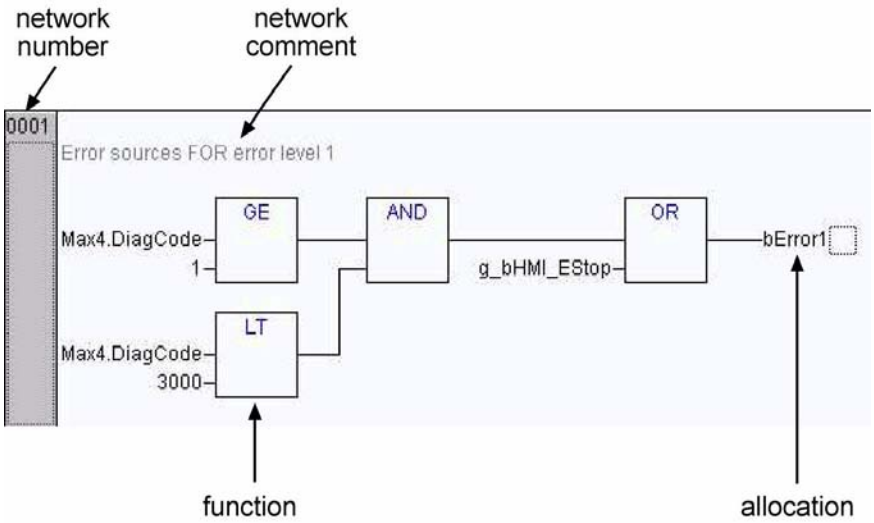



*Illustration 4-11: Réseau en langage en blocs fonctionnels*

**Langage à contacts (LD)**

La programmation en LD (de l'anglais « Ladder Diagram ») est le moyen le plus simple de donner à un automate une fonctionnalité bien définie. Le langage à contacts s'avère particulièrement intéressant lorsque l'automate doit être utilisé en remplacement d'une logique câblée ou pour programmer des fonctions de base comme des verrouillages de démarrage. Il est aussi parfaitement adapté à la programmation d'opérations logiques ET/OU complexes. En raison de sa simplicité, il est d'ailleurs devenu le langage préféré de certains pays, comme les Etats-Unis. Comme la norme prévoit que les blocs complexes (écrits dans d'autres langages) peuvent être lancés à l'aide d'une entrée Enable, il est possible de commander des constructions puissantes à l'aide de la logique à contacts.
La représentation graphique simule un flux de courant amené par une « barre d'alimentation gauche » et franchissant des contacts d'entrée (représentés par des variables) via des circuits aboutissant à des actionneurs de sortie (bobinages), également représentés par des variables. Les sauts et les retours au début sont autorisés.

Le langage à contacts est également un langage de programmation orienté graphique, qui se rapproche du principe du schéma électrique.

S'il convient bien à la programmation de séquences logiques, il permet aussi de réaliser des réseaux comme dans le langage FBD. Il peut donc être utilisé pour commander l'appel d'autres blocs. Cette question sera abordée ultérieurement de manière plus détaillée.

Le schéma à contacts est constitué d'une suite de réseaux. Un réseau est délimité à gauche et à droite par une barre d'alimentation verticale gauche et droite. Le circuit situé entre ces deux barres se compose de contacts, de bobinages et de lignes de liaison.

Chaque réseau comprend, dans sa partie gauche, une succession de contacts qui transmettent de gauche à droite l'état « ON » ou « OFF » ; ces états correspondent respectivement aux valeurs booléennes TRUE et FALSE. A chaque contact est affectée une variable booléenne. Lorsque cette variable a la valeur TRUE, l'état est transmis de gauche à droite via la ligne de liaison ; dans le cas contraire, la liaison de droite prend la valeur « OFF ».

Exemple de réseau dans le langage à contacts tel qu'il se présenterait dans l'EPAS-4 :

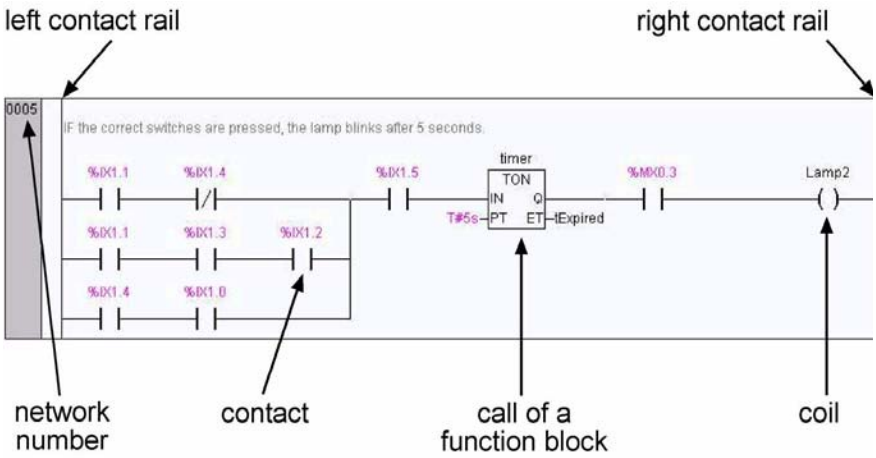



*Illustration 4-12: Réseau en langage à contacts*

Contact

Chaque réseau en LD est constitué, dans sa partie gauche, d'un réseau de contacts (les contacts sont représentés par deux barres parallèles : | |), qui transmettent de gauche à droite l'état « ON » ou « OFF ».

Ces états correspondent aux valeurs booléennes TRUE et FALSE. A chaque contact est affectée une variable booléenne. Lorsque cette variable a la valeur TRUE, l'état est transmis de gauche à droite via la ligne de liaison ; dans le cas contraire, la liaison de droite prend la valeur « OFF ».

Si les contacts sont couplés en parallèle, au moins une des branches parallèles doit transmettre la valeur « ON » pour que le branchement parallèle transmette l'état « ON ». Si les contacts sont couplés en série, tous les contacts doivent transmettre la valeur « ON » pour que le dernier contact transmette l'état « ON ». Cela correspond donc à une mise en parallèle ou en série dans un schéma électrique.

Un contact peut également être négatif ; le symbole de contact comporte alors une barre oblique : |/|. La valeur de la liaison est dans ce cas transmise si la variable a la valeur FALSE.

PDM_UserMan_IEC_fr.fm

Bobinage

La partie droite d'un réseau LD comporte un nombre quelconque de bobinages, qui sont représentés par des parenthèses : ( ). Ceux-ci ne peuvent être que couplés en parallèle. Un bobinage transmet la valeur des liaisons de gauche à droite et copie cette valeur dans la variable booléenne associée. Les valeurs « ON » ou « OFF » peuvent être présentes sur la ligne d'entrée (correspondant aux variables booléennes TRUE ou FALSE).

Comme les contacts, les bobinages peuvent être négativés (dans l'exemple, le contact COMMUT1 et le bobinage %QX3.0 sont négativés). Lorsqu'un bobinage est négativé (le symbole de bobinage comporte alors une barre oblique : (/)), il copie la valeur inversée dans la variable booléenne associée. Lorsqu'un contact est négativé, il ne commute que si la variable associée a la valeur FALSE.

Blocs fonctionnels en langage à contacts

Outre des contacts et des bobines, vous pouvez également insérer dans un schéma à contacts des blocs fonctionnels et des programmes. Ceux-ci doivent posséder une entrée et une sortie associées à des valeurs booléennes et peuvent être utilisés aux mêmes emplacements que les contacts (dans la partie gauche du réseau LD, par exemple).

Bobinage Set/Reset

Les bobinages peuvent également être définis comme bobinages Set ou Reset. Un bobinage Set (reconnaissable au « S » situé dans le symbole du bobinage : (S)) n'écrase jamais la valeur TRUE dans la variable booléenne associée. Si la variable a été positionnée une fois à TRUE, elle conserve donc cet état.

Un bobinage Reset (reconnaissable au « R » situé dans le symbole du bobinage : (R)) n'écrase jamais la valeur FALSE dans la variable booléenne associée. Si la variable a été positionnée une fois à FALSE, elle conserve donc cet état.

LD comme FBD

Lors de la programmation en LD, il peut arriver que vous ayez besoin d'utiliser le résultat de la commutation d'un contact pour la commande d'autres blocs. Vous avez alors la possibilité de déposer, à l'aide de bobinages, le résultat dans une variable globale qui sera utilisée plus tard. Mais vous pouvez aussi insérer directement dans votre réseau LD un appel éventuel en utilisant un bloc doté d'une entrée EN.

Ces blocs sont des opérandes, des fonctions, des programmes ou des blocs fonctionnels tout à fait normaux, qui comportent cependant une entrée supplémentaire marquée EN. L'entrée EN est toujours du type BOOL. Le bloc comportant l'entrée EN est évalué lorsque EN prend la valeur TRUE.

Le bloc EN est couplé en parallèle avec les bobinages tandis que l'entrée EN est connectée à la ligne de liaison entre les contacts et les bobinages. Lorsque l'information « ON » est transmise via cette ligne, le bloc est évalué tout à fait normalement.

A partir de tels blocs EN, il est possible de programmer des réseaux comme dans le langage FBD.

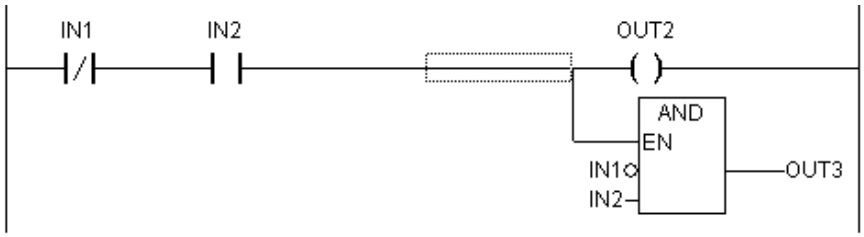



*Illustration 4-13: Partie d'un réseau LD comportant un bloc EN*

PDM_UserMan_IEC_fr.fm

## 4.2.5 Opérateurs

Contrairement aux fonctions standards, les opérateurs IEC sont implicitement connus dans l'ensemble du projet. Lors de la programmation des blocs, les opérateurs s'utilisent de la même manière que les fonctions. Les opérateurs admis par le système sont énumérés dans le tableau ci-dessous.

### Vue d'ensemble

Les opérateurs servent à relier les opérandes dans les expressions.

| Opérateur/ Symbole | Description | Explications |
|---|---|---|
| [] | Début et fin pour :<br>1. Indice de tableau (accès à un élément de tableau)<br>2. Longueur de cordon de caractères (dans la déclaration) | |
| ( ... ) | Mise entre parenthèses | A:= (5*2) + (4/2)<br>-> A = 12 |
| Nom de la fonction (liste de paramètres) | Evaluation de la fonction | ADD (A, B, C) |
| EXPT<br>** | Exponentiation<br>(** non géré par le système) | A:= 2**3<br>-> A = 8 |
| - | Négation | A:= -5<br>-> A = -5 |
| NOT | Complément | |
| MUL<br>* | Multiplication | A:= 5*2<br>-> A = 10 |
| DIV<br>/ | Division | A:= 4/2<br>-> A = 2 |
| MOD | Modulo<br>(non admis pour REAL et LREAL) | A:= 12 MOD 10<br>-> A = 2 |
| ADD<br>+ | Addition | A:= 3+4<br>-> A = 7 |
| SUB<br>- | Soustraction | A:= 4-2<br>-> A = 2 |
| > | Opérateur de comparaison supérieur à | |
| < | Opérateur de comparaison inférieur à | |
| >= | Opérateur de comparaison supérieur ou égal à | |

| Opérateur/ Symbole | Description | Explications |
|---|---|---|
| <= | Opérateur de comparaison inférieur ou égal à | |
| = | Opérateur de comparaison égal à | |
| <> | Opérateur de comparaison différent de | |
| AND<br>& | ET booléen | |
| XOR | OU exclusif booléen | |
| OR | OU booléen | |
| MOVE<br>:= | Affectation<br>1. Opérateur pour affectation de valeur initiale<br>2. Opérateur de liaison d'entrée (affectation de paramètre courant à un paramètre formel lors d'un appel d'UOP)<br>3. Opérateur d'instruction | |

*Tableau 4-14: Opérateurs*

Les opérateurs de même niveau de priorité sont évalués de gauche à droite.

| Priorité | Opération | Opérateur |
|---|---|---|
| Maximale | | [] |
| | Mise entre parenthèses | ( ... ) |
| | Evaluation de fonction | Nom de la fonction (liste de paramètres) |
| | Exponentiation | ** |
| | Négation | - |
| | Complément | NOT |
| | Multiplication | * |
| | Division | / |
| | Modulo | MOD |
| | Addition | + |
| | Soustraction | - |
| | Comparaison | >, <, >=, <= |
| | Egalité | = |
| | Inégalité | <> |
| | ET | AND, & |
| | OU exclusif | XOR |
| | OU | OR |
| Minimale | Affectation | := |

*Tableau 4-15: Opérateurs : priorité et associativité*

PDM_UserMan_IEC_fr.fm

Autres opérateurs :

| Nom | Fonction | Description |
|---|---|---|
| *_TO_** | | |
| TRUNC | Conversion d'une valeur REAL en une valeur INT | |
| BCD_TO_** | Conversion d'un valeur BCD en une valeur INT | |
| *_TO_BCD | Conversion par ex. d'une valeur INT en une valeur BCD | |

*Tableau 4-16: Opérateurs de conversion de types*

* Type de donnée de l'entrée, par ex. REAL
** Type de donnée de la sortie, par ex. INT

Autres opérateurs :

| Nom | Fonction | Description |
|---|---|---|
| ABS | Valeur absolue | F:= \| IN \| |
| SQRT | Racine carrée | F:= √IN |
| LN | Logarithme naturel | F:= $\log_e$ (IN) |
| LOG | Logarithme en base 10 | F:= $\log_{10}$ (IN) |
| EXP | Exponentielle naturelle | F:= eIN |
| SIN | Sinus d'entrée en radians | F:= SIN (IN) |
| COS | Cosinus d'entrée en radians | F:= COS (IN) |
| TAN | Tangente d'entrée en radians | F:= TAN (IN) |
| ASIN | Arc sinus principal | F:= ARCSIN (IN) |
| ACOS | Arc cosinus principal | F:= ARCCOS (IN) |
| ATAN | Arc tangente principal | F:= ARCTAN (IN) |
| EXPT | Exponentiation d'une variable par une autre | F:= IN1IN2 |

*Tableau 4-17: Opérateurs numériques*

**Opérateurs arithmétiques**

ADD

Addition de variables de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL ou LREAL.

On peut également additionner deux variables TIME. La somme est aussi un temps (par ex. t#45s + t#50s = t#1m35s)

Exemple en IL :

```
LD   7
ADD  2,4,7
ST   var1
```

Exemple en ST :

```
var1 := 7+2+4+7;
```

MUL

Multiplication de variables de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL ou LREAL.

Exemple en IL :

```
LD   7
MUL  2,4,7
ST   var1
```

Exemple en ST :

```
var1 := 7*2*4*7;
```

SUB

Soustraction d'une variable de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL ou LREAL d'une autre variable de l'un de ces types.

On peut également soustraire une variable TIME d'une autre variable TIME. Le résultat est aussi de type TIME. Veillez à ne pas définir de valeurs TIME négatives.

Exemple en IL :

```
LD   7
SUB  8
ST   var1
```

Exemple en ST :

```
var1 := 7-2;
```

DIV

Division d'une variable de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL ou LREAL par une autre variable de l'un de ces types.

Exemple en IL :

```
LD 8
DIV2
STvar1
```

Exemple en ST :

```
var1 := 8/2;
```

PDM_UserMan_IEC_fr.fm

MOD

Division modulo d'une variable de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL ou LREAL par une autre variable de l'un de ces types. Cette fonction fournit comme résultat le reste entier de la division.

Exemple en IL :

```
LD   9
MOD  2
ST   var1 (* Le résultat est 1 *)
```

Exemple en ST:

```
var1 := 9 MOD 2;
```

INDEXOF

Cette fonction fournit comme résultat l'indice interne d'un bloc.

Exemple en ST :

```
var1 := INDEXOF(bloc2);
```

SIZEOF

Cette fonction fournit comme résultat le nombre d'octets que nécessite le type de donnée indiqué.

Exemple en IL :

```
arr1:ARRAY[0..4] OF INT;
var1:=INT;
LD   arr1
SIZEOF
ST   var1   (* Le résultat est 10 *)
```

Exemple en ST :

```
var1: INT;
pt := ADR(pt^) + SIZEOF(var1);
```

**Opérateurs de cordons de bits**

AND

ET effectué entre bits de même poids d'opérandes de type BOOL, BYTE, WORD ou DWORD.

Exemple en IL :

```
var1 :BYTE;
LD   2#1001_0011
AND  2#1000_1010
ST   var1 (* Le résultat est 2#1000_0010 *)
```

Exemple en ST :

```
var1 := 2#1001_0011 AND 2#1000_1010
```

OR

OU effectué entre bits de même poids d'opérandes de type BOOL, BYTE, WORD ou DWORD.

Exemple en IL :

```
var1 :BYTE;
LD   2#1001_0011
OR   2#1000_1010
ST   var1 (* Le résultat est 2#1001_1011 *)
```

Exemple en ST :

```
Var1 := 2#1001_0011 OR 2#1000_1010
```

XOR

OU exclusif effectué entre bits de même poids d'opérandes de type BOOL, BYTE, WORD ou DWORD.

Exemple en IL :

```
Var1 :BYTE;
LD   2#1001_0011
XOR  2#1000_1010
ST   Var1 (* Le résultat est 2#0001_1001 *)
```

Exemple en ST :

```
Var1 := 2#1001_0011 XOR 2#1000_1010
```

NOT

Complémentation bit par bit d'un opérande de type BOOL, BYTE, WORD ou DWORD.

Exemple en IL :

```
Var1 :BYTE;
LD2#1001_0011
NOT
STVar1 (* Le résultat est 2#0110_1100 *)
```

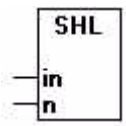
Exemple en ST :

```
Var1 := NOT 2#1001_0011
```

**Opérateurs de décalage de bits**

Les opérateurs suivants sont représentés à l'aide d'un bloc FBD.

SHL




Décalage à gauche des bits d'un opérande : A:= SHL (IN, N)

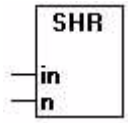
A, IN et N doivent être de type BYTE, WORD ou DWORD. IN est décalé de N bits vers la gauche tandis que les positions libérées sont remplies par des zéros.

Exemple :

```
LD   1
SHL  1
ST   Var1 (* Le résultat est 2 *)
```

SHR

PDM_UserMan_IEC_fr.fm



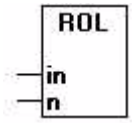

Décalage à droite des bits d'un opérande : A:= SHR (IN, N)

A, IN et N doivent être de type BYTE, WORD ou DWORD. IN est décalé de N bits vers la droite tandis que les positions libérées sont remplies par des zéros.

Exemple :

```
LD   32
SHL  2
ST   Var1(* Le résultat est 8 *)
```

ROL



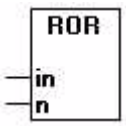

Rotation à gauche des bits d'un opérande : A:= ROL (IN, N)

A, IN et N doivent être de type BYTE, WORD ou DWORD. IN est décalé N fois d'une position vers la gauche tandis que les bits sortants sont renvoyés à droite vers les positions libérées.

Exemple :

```
Var1 :BYTE;
LD2#1001_0011
ROL3
STVar1 (* Le résultat est 2#1001_1100 *)
```

ROR




Rotation à droite des bits d'un opérande : A:= ROR (IN, N)

A, IN et N doivent être de type BYTE, WORD ou DWORD. IN est décalé N fois d'une position vers la droite tandis que les bits sortants sont renvoyés à gauche vers les positions libérées.

Exemple :

```
Var1 :BYTE;
LD   2#1001_0011
ROR  3
ST   Var1 (* Le résultat est 2#0111_0010 *)
```

PDM_UserMan_IEC_fr.fm

### Opérateurs de sélection

Toutes les opérations de sélection peuvent également être effectuées sur des variables. Pour plus de clarté, nous nous limiterons cependant, dans les exemples suivants, à l'utilisation de constantes.

SEL

Sélection binaire.

OUT := SEL(G, IN0, IN1) signifie :

OUT := IN0 if G=FALSE;

OUT := IN1 if G=TRUE.

IN0, IN1 et OUT peuvent être de type quelconque, G doit être de type BOOL. Le résultat de la sélection est IN0 si G a la valeur FALSE et IN1 si G a la valeur TRUE.

Exemple en IL :

```
LD   TRUE
SEL  3,4
ST   Var1 (* Le résultat est 4 *)
LD   FALSE
SEL  3,4
ST   Var1 (* Le résultat est 3 *)
```

MAX

Sélection du maximum. Détermine, parmi deux opérandes, celui qui a la plus forte valeur.

OUT := MAX(IN0, IN1)

IN0, IN1 et OUT peuvent être de type quelconque.

Exemple en IL :

```
LD   90
MAX  30
MAX  40
MAX  77
ST   Var1 (* Le résultat est 90 *)
```

MIN

Sélection du minimum. Détermine, parmi deux opérandes, celui qui a la plus faible valeur.

OUT := MIN(IN0, IN1)

IN0, IN1 et OUT peuvent être de type quelconque.

Exemple en IL :

```
LD   90
MIN  30
MIN  40
MIN  77
ST   Var1 (* Le résultat est 30 *)
```

LIMIT

Limitation

OUT := LIMIT(Min, IN, Max) signifie :

OUT := MIN (MAX (IN, Min), Max)

PDM_UserMan_IEC_fr.fm

Max définit la valeur limite supérieure du résultat, Min la valeur limite inférieure. Si la valeur IN dépasse la limite supérieure Max, LIMIT fournit le résultat Max. Si la valeur IN est inférieure à Min, le résultat est Min.

IN et OUT peuvent être de type quelconque.

Exemple en IL :

```
LD     90
LIMIT  30,80
ST     Var1 (* Le résultat est 80 *)
```

MUX

Multiplexeur

OUT := MUX(K, IN0,...,INn) signifie :

OUT := INK.

IN0, ...,INn et OUT peuvent être de type quelconque. K doit être de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT ou UDINT. MUX sélectionne parmi un ensemble de valeurs la Kième.

Exemple en IL :

```
LD   0
MUX  30,40,50,60,70,80
ST   Var1 (* Le résultat est 30 *)
```

**Opérateurs de comparaison**

GT

Supérieur à

Cet opérateur booléen fournit le résultat TRUE si le premier opérande est supérieur au deuxième. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

Exemple en IL :

```
LD   20
GT   30
ST   Var1 (* Le résultat est FALSE *)
```

Exemple en ST :

```
VAR1 := 20 > 30 > 40 > 50 > 60 > 70;
```

LT

Inférieur à

Cet opérateur booléen fournit le résultat TRUE si le premier opérande est inférieur au deuxième. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

Exemple en IL :

```
LD   20
LT   30
ST   Var1 (* Le résultat est TRUE *)
```

Exemple en ST :

```
VAR1 := 20 < 30;
```

LE

Inférieur ou égal à

Cet opérateur booléen fournit le résultat TRUE si le premier opérande est inférieur ou égal au deuxième. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

Exemple en IL :

```
LD   20
LE   30
ST   Var1 (* Le résultat est TRUE *)
```

Exemple en ST :

```
VAR1 := 20 <= 30;
```

GE

Supérieur ou égal à

Cet opérateur booléen fournit le résultat TRUE si le premier opérande est supérieur ou égal au deuxième. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

Exemple en IL :

```
LD   60
GE   40
ST   Var1 (* Le résultat est TRUE *)
```

Exemple en ST :

```
VAR1 := 60 >= 40;
```

EQ

Egal à

Cet opérateur booléen fournit l e résultat TRUE si les opérandes sont égaux. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

Exemple en IL :

```
LD   40
EQ   40
ST   Var1 (* Le résultat est TRUE *)
```

Exemple en ST :

PDM_UserMan_IEC_fr.fm

```
VAR1 := 40 = 40;
```

NE

Différent de

Cet opérateur booléen fournit le résultat TRUE si les opérandes sont différents. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, U-INT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

Exemple en IL :

```
LD   40
NE   40
ST   Var1 (* Le résultat est FALSE *)
```

Exemple en ST :

```
VAR1 := 40 <> 40;
```

**Opérateurs d'adresse**

ADR

Fonction d'adresse

ADR fournit l'adresse de son argument dans un DWORD. Cette adresse peut être envoyée à des fonctions constructeur pour y être traitée comme un pointeur ou être affectée à un pointeur à l'intérieur du projet.

Exemple en IL :

```
LD   var1
ADR
ST   var2
man_fun1
```

Opérateur de contenu

Le déréférencement d'un pointeur s'effectue à l'aide de l'opérateur de contenu « ^ » placé après l'identificateur du pointeur.

Exemple en ST :

```
pt:POINTER TO INT;
var_int1:INT;
var_int2:INT;
pt := ADR(var_int1);
var_int2:=pt^;
```

**Opérateur d'appel**

CAL

Appel d'un bloc fonctionnel

CAL permet d'appeler en IL l'instance d'un bloc fonctionnel. Le nom de l'instance du bloc fonctionnel est suivi, entre parenthèses, d'une affectation de valeurs aux variables d'entrée du bloc fonctionnel.

Exemple :

Appel de l'instance Inst d'un bloc fonctionnel avec affectation des valeurs 0 ou TRUE aux variables d'entrée Par1, Par2.

```
CAL INST(PAR1 := 0, PAR2 := TRUE)
```

**Opérateurs de conversion de type**

La conversion implicite d'un type de donnée de poids fort en un type de donnée de poids inférieur (de INT en BYTE ou de DINT en WORD, par exemple) n'est pas autorisée. Si l'application l'exige, on doit faire appel à des fonctions de conversion de type spéciales. Il est en principe possible de convertir n'importe quel type de donnée élémentaire en n'importe quel autre type de donnée élémentaire.

Syntaxe :

<Type.elem1>_TO_<Type.elem2>

Conversions BOOL_TO

Conversion de type BOOL en un autre type :

Avec des types de données numériques, le résultat est 1 si l'opérande est TRUE et 0 si l'opérande est FALSE.

Avec le type STRING, le résultat est 'TRUE' ou 'FALSE'.

Exemples en ST :

```
i:=BOOL_TO_INT(TRUE);
(* -> Le résultat est 1 *)
str:=BOOL_TO_STRING(TRUE);
(* -> Le résultat est 'TRUE' *)
t:=BOOL_TO_TIME(TRUE);
(* -> Le résultat est T#1ms *)
tof:=BOOL_TO_TOD(TRUE);
(* -> Le résultat est TOD#00:00:00.001 *)
dat:=BOOL_TO_DATE(FALSE);
(* -> Le résultat est D#1970-01-01 *)
dandt:=BOOL_TO_DT(TRUE);
(* -> Le résultat est DT#1970-01-01-00:00:01 *)
```

Conversions TO_BOOL

Conversion d'un type en un type BOOL :

Le résultat est TRUE si l'opérande est différent de 0. Le résultat est FALSE si l'opérande est égal à 0.

Avec le type STRING, le résultat est TRUE si l'opérande est 'TRUE', sinon le résultat est FALSE.

Exemples en ST :

```
b := BYTE_TO_BOOL(2#11010101);
(* -> Le résultat est TRUE *)
b := INT_TO_BOOL(0);
(* -> Le résultat est FALSE *)
b := TIME_TO_BOOL(T#5ms);
(* -> Le résultat est TRUE *)
b := STRING_TO_BOOL('TRUE');
(* -> Le résultat est TRUE *)
```

Conversions entre types numériques entiers

Conversion d'un type numérique entier en un autre type numérique :

PDM_UserMan_IEC_fr.fm

Lors de la conversion de types de poids fort en types de poids inférieur, des informations peuvent être perdues. Si le nombre à convertir dépasse la limite de la plage, les premiers octets du nombre ne sont pas pris en compte.

Exemple en ST :

```
si := INT_TO_SINT(4223); (* Le résultat est 127 *)
```

Si vous mémorisez le nombre entier 4223 (16#107f en représentation hexadécimale) dans une variable SINT, celle-ci contient le nombre 127 (16#7f en représentation hexadécimale).

Exemple en IL :

```
LD   2
INT_TO_REAL
MUL  3.5
```

Conversions REAL_TO/LREAL_TO

Conversion du type REAL ou LREAL en un autre type :

La valeur est arrondie à l'entier supérieur ou inférieur et convertie dans le type spécifié. Les types STRING, BOOL, REAL et LREAL font exception.

Lors de la conversion de types de poids fort en types de poids inférieur, des informations peuvent être perdues.

Exemple en ST :

```
i := REAL_TO_INT(1.5); (* Le résultat est 2 *)
j := REAL_TO_INT(1.4); (* Le résultat est 1 *)
```

Exemple en IL :

```
LD   2.7
REAL_TO_INT
GE   %MW8
```

Conversions TIME_TO/TIME_OF_DAY

Conversion du type TIME ou TIME_OF_DAY en un autre type :

En interne, le temps est mémorisé dans un DWORD en millisecondes (à partir de 00:00 heure avec TIME_OF_DAY). Cette valeur est ensuite convertie.

Lors de la conversion de types de poids fort en types de poids inférieur, des informations peuvent être perdues.

Avec le type STRING, le résultat est la constante de temps.

Exemple en ST :

```
str :=TIME_TO_STRING(T#12ms);
(* -> Le résultat est 'T#12ms' *)
dw:=TIME_TO_DWORD(T#5m);
(* -> Le résultat est 300000 *)
si:=TOD_TO_SINT(TOD#00:00:00.012);
(* -> Le résultat est 12 *)
```

PDM_UserMan_IEC_fr.fm

Conversions DATE_TO/DT_TO

Conversion du type DATE ou DATE_AND_TIME en un autre type :

En interne, la date est mémorisée dans un DWORD en secondes à partir du 1er janvier 1970. Cette valeur est ensuite convertie.

Lors de la conversion de types de poids fort en types de poids inférieur, des informations peuvent être perdues.

Avec le type STRING, le résultat est la constante de date.

Exemples en ST :

```
b :=DATE_TO_BOOL(D#1970-01-01);
(* -> Le résultat est FALSE *)
i :=DATE_TO_INT(D#1970-01-15);
(* -> Le résultat est 29952 *)
byt :=DT_TO_BYTE(DT#1970-01-15-05:05:05);
(* -> Le résultat est 129 *)
str:=DT_TO_STRING(DT#1998-02-13-14:20);
(* -> Le résultat est 'DT#1998-02-13-14:20' *)
```

Conversions STRING_TO

Conversion du type STRING en un autre type :

L'opérande de type STRING doit avoir une valeur admise par le type de destination, sinon le résultat est 0.

Exemples en ST :

```
b :=STRING_TO_BOOL('TRUE');
(* -> Le résultat est TRUE *)
w :=STRING_TO_WORD('abc34');
(* -> Le résultat est 0 *)
t :=STRING_TO_TIME('T#127ms');
(* -> Le résultat est T#127ms *)
```

TRUNC

Conversion du type REAL en type INT. Seule la partie entière du nombre est conservée.

Lors de la conversion de types de poids fort en types de poids inférieur, des informations peuvent être perdues.

Exemples en ST :

```
i:=TRUNC(1.9); (* Le résultat est 1 *).
i:=TRUNC(-1.4); (* Le résultat est -1 *).
```

Exemple en IL :

```
LD   2.7
TRUNC
GE   %MW8
```

PDM_UserMan_IEC_fr.fm

**Opérateurs numériques**

ABS

Fournit la valeur absolue d'un nombre. ABS(-2) donne 2.

SQRT

Fournit la racine carrée d'un nombre.

LN

Fournit le logarithme naturel d'un nombre.

LOG

Fournit le logarithme en base 10 d'un nombre.

EXP

Fournit la fonction exponentielle.

SIN

Fournit le sinus d'un nombre.

COS

Fournit le cosinus d'un nombre.

TAN

Fournit la tangente d'un nombre.

ASIN

Fournit l'arc sinus (fonction inverse de sinus) d'un nombre.

ACOS

Fournit l'arc cosinus (fonction inverse de cosinus) d'un nombre.

ATAN

Fournit l'arc tangente (fonction inverse de la tangente) d'un nombre.

EXPT

Exponentiation d'une variable par une autre :

```
OUT = IN1IN2.
```

OUT, IN1 et IN2 peuvent être de type BYTE, WORD, DWORD, INT, DINT ou REAL.

Exemple en IL :

```
LD   7
EXPT 2
ST   var1 (* Le résultat est 49 *)
```

Exemple en ST :

```
var1 := EXPT (7,2);
```

### 4.2.6 Opérandes

Dans le PacDrive™, les opérandes peuvent être des constantes, des variables, des adresses et, éventuellement, des appels de fonctions.

**Constantes**

Constantes BOOL

Les constantes BOOL ont les valeurs logiques TRUE et FALSE.

Constantes TIME

Des constantes TIME peuvent être déclarées dans l'EPAS-4. Celles-ci s'utilisent en particulier pour le paramétrage des temporisateurs de la bibliothèque standard. Une constante TIME commence toujours par un « t » ou « T » (ou « time » ou « TIME » dans la forme longue) suivi d'un dièse « # ».

La déclaration de temps proprement dite est placée après ces préfixes. Elle peut se composer de jours (d), d'heures (h), de minutes (m), de secondes (s) et de millisecondes (ms). Les unités doivent être indiquées par ordre de grandeur croissant (d avant h avant m avant s avant m avant ms), mais ne doivent pas nécessairement être toutes présentes.

Exemples de constantes TIME correctes dans une affectation ST :

```
TIME1 := T#14ms;
TIME1 := T#100S12ms;
(*>Le dépassement de l'unité de plus fort poids est
admis *)
TIME1 := t#12h34m15s;
```

Les libellés suivants ne seraient pas corrects :

```
TIME1 := t#5m68s;
(* -> Dépassement d'une unité de poids faible *)
TIME1 := 15ms;
(* -> Il manque le préfixe T# *)
TIME1 := t#4ms13d;
(* -> Ordre des unités incorrect *)
```

Constantes DATE

Ce type permet de spécifier des dates. Une constante DATE se déclare à l'aide du préfixe « d », « D », « DATE » ou « date », suivi d'un « # ». Vous pouvez ensuite entrer une date quelconque dans l'ordre Année-Mois-Jour.

Exemples :

```
DATE#1996-05-06
d#1972-03-29
```

Constantes TIME_OF_DAY

Ce type permet de mémoriser des heures. Une déclaration TIME_OF_DAY commence par « tod# », « TOD# », « TIME_OF_DAY# » ou « time_of_day# ».Vous pouvez ensuite entrer une heure dans l'ordre : heure, minutes, secondes. Les secondes peuvent être indiquées en tant que nombres réels. Les fractions de secondes sont admises.

PDM_UserMan_IEC_fr.fm

Exemples :

```
TIME_OF_DAY#15:36:30.123
tod#00:00:00
```

Constantes DATE_AND_TIME

Les constantes de date et d'heure peuvent également être combinées en constantes DATE_AND_TIME. Les constantes DATE_AND_TIME commencent par « dt# », « DT# », « DATE_AND_TIME# » ou « date_and_time# ». L'indication de la date est suivie d'un trait d'union, puis de l'heure.

Exemples :

```
DATE_AND_TIME#1996-05-06-15:36:30
dt#1972-03-29-00:00:00
```

Constantes numériques

Les valeurs numériques peuvent être représentées sous forme binaire, octale, décimale et hexadécimale. Si une valeur entière n'est pas un nombre décimal, sa base, suivie d'un dièse (#), doit être écrite avant la constante entière. En hexadécimal, les valeurs numériques des nombres 10 à 15 sont représentées comme d'habitude à l'aide des lettres A-F.

Les traits de soulignement à l'intérieur d'une valeur numérique sont autorisés.

Exemples :

```
14          (Nombre décimal)
2#1001_0011 (Nombre binaire)
8#67        (Nombre octal)
16#A        (Nombre hexadécimal)
```

Ces valeurs numériques peuvent être de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL ou LREAL.

Les conversions implicites de types de poids fort en types de poids inférieur ne sont pas autorisées. Il n'est donc pas possible d'utiliser directement une variable DINT en tant que variable INT. On doit, pour ce faire, utiliser les fonctions de conversion de standard.lib

Constantes REAL/LREAL

Les constantes REAL et LREAL peuvent être représentés sous forme de fractions décimales et avec un exposant. On utilise dans ce cas la notation américaine avec point.

Exemples :

```
7.4 au lieu de 7,4
1.64e+009 au lieu de 1,64e+009
```

Constantes STRING

Les constantes STRING (cordons de caractères) sont composées d'une suite de caractères quelconques. Elles sont délimitées par des apostrophes. Elles peuvent contenir des espaces et des signes d'accentuation. Ils sont traités de la même manière que les autres caractères.

Dans les cordons de caractères, la combinaison à trois caractères du signe dollar ($) suivi de deux chiffres hexadécimaux est interprétée comme la représentation hexadécimale du code de caractères à huit bits. Les combinaisons à deux caractères commençant par le signe dollar doivent être interprétées comme suit lorsqu'elles apparaissent dans des cordons de caractères :

$$ Signe dollar

$' 'Apostrophe

$L ou $lSaut de ligne

$N ou $nNouvelle ligne

$P ou $pChangement de page

$R or $rRetour du curseur

$T or $tTabulation

Exemples :

```
'OK, ça marche !'
'Michaël'
':-)'
```

**Variables**

Les variables sont déclarées soit localement dans la partie déclaration d'un bloc, soit globalement dans les listes de variables globales.

Les identificateurs des variables ne doivent contenir aucun espace ni caractère accentué ; ils ne doivent pas non plus être déclarés en double et être identiques à des mots clés. Aucune distinction n'est faite entre majuscules et minuscules : VAR1, Var1 et var1 ne constituent donc pas des variables différentes. Les traits de soulignement ont une signification particulière : par exemple, « A_BCD » et « AB_CD » sont interprétés comme des identificateurs différents. Les traits de soulignement multiples en tête ou intégrés ne sont pas autorisés. Les 32 premiers caractères sont significatifs.

Les variables peuvent être utilisées partout où le type déclaré le permet.

Vous pouvez appeler les variables disponibles via l'aide d'entrée.

Drapeaux système

Les drapeaux système sont des variables implicitement déclarées qui sont dépendantes de l'automate. Pour connaître les drapeaux spécifiques à votre système, choisissez la commande 'INSERT OPERAND'. Dans la boîte de dialogue qui s'affiche, sélectionnez la catégorie System Variable.

PDM_UserMan_IEC_fr.fm

Accès à des variables de tableaux, structures et blocs

On accède aux éléments des tableaux bidimensionnels à l'aide de la syntaxe suivante :

```
<NomTableau>[Indice1, Indice2}]
```

On accède aux variables des structures à l'aide de la syntaxe suivante :

```
<NomStructure>.<NomVariable>
```

On accède aux variables des blocs fonctionnels et des programmes à l'aide de la syntaxe suivante :

```
<NomBloc>.<NomVariable>
```

**Adresses**

Adresse

La représentation directe d'emplacements mémoire individuels s'effectue à l'aide d'un symbole spécial formé par l'enchaînement d'un signe de pourcentage « % », d'un préfixe d'emplacement, d'un préfixe de taille et d'un ou plusieurs entiers non signés, séparés par des points.

Les préfixes d'emplacement suivants sont gérés par le système :

I Entrée

Q Sortie

M Mémoire interne

Les préfixes de taille suivants sont gérés par le système :

X Un seul bit

AucunUn seul bit

B Octet (8 bits)

W Mot (16 bits)

D Double mot (32 bits)

Exemples :

%QX75 et %Q75 Bit de sortie 75
%IW215 Mot d'entrée 215
%QB7 Octet de sortie 7
%MD48 Double mot à l'emplacement 48 de la mémoire interne
%IW2.5.7.1dépendant de la configuration de l'automate

La validité d'une adresse dépend de la configuration courante de l'automate dans le programme.

Mémoires internes

Pour accéder aux mémoires internes, on peut utiliser tous les préfixes de taille gérés.

L'adresse %MD48, par exemple, permettrait d'adresser les octets n° 192, 193, 194 et 195 dans la plage de mémoires internes (48 * 4 = 192). Le premier octet est l'octet n° 0.

On accède de la même manière aux mots, aux octets, et même aux bits : %MX5.0, par exemple, permet d'adresser le premier bit du cinquième mot (les bits sont généralement mémorisés dans des mots).

### Fonctions

En langage ST, un appel de fonction peut également être utilisé comme opérande.

Exemple :

```
Resultat := Fct(7) + 3;
```

PDM_UserMan_IEC_fr.fm

### 4.2.7 Bibliographie

La norme IEC 61131 est la norme applicable aux systèmes d'automates programmables industriels. Elle se compose essentiellement de 5 parties et de 2 rapports techniques. La partie de loin la plus importante de la norme est la partie 3 : Langages de programmation.

Cette partie décrit un ensemble de langages de programmation performants utilisables pour différentes applications.

Plusieurs ouvrages ont été publiés afin d'aider l'utilisateur à mieux comprendre la norme IEC 61131-3. Nous voudrions en citer quelques-uns :

- SPS-Standard IEC 1131
  Karl-Heinz John, Michael Tiegelkamp
  ISBN 3-540-58635-0
- SPS-Programmierung mit IEC 1131-3
  Neumann / Grötsch / Lubkoll / Simon
  ISBN 3-486-23348-3
- SPS-Programmierung nach IEC 1131-3
  (recommandé pour approfondir le thème)
  Wellers
  ISBN 3-464-48062-3
- Moderne Programmiertechnik für Automatisierungssysteme
  Peter Wratil
  ISBN 3-8023-1575-8
- Grundkurs IEC 1131
  (Exemples de programmation pour l'automatisation de processus)
  Karl Pusch
  ISBN 3-8023-1807-2

# 4.3 Particularités du PacDrive™

## 4.3.1 Principe de fonctionnement

Le système PacDrive est un système multitâches. Pour éviter les problèmes d'accès aux entrées et sorties globales du système, le système ne crée pas d'image des entrées et des sorties pour chaque tâche.
Les entrées et sorties sont dans ce cas lues ou positionnées dès qu'une instruction accédant aux entrées ou aux sorties est traitée. Ce mode de traitement permet en outre d'accroître la vitesse de traitement.

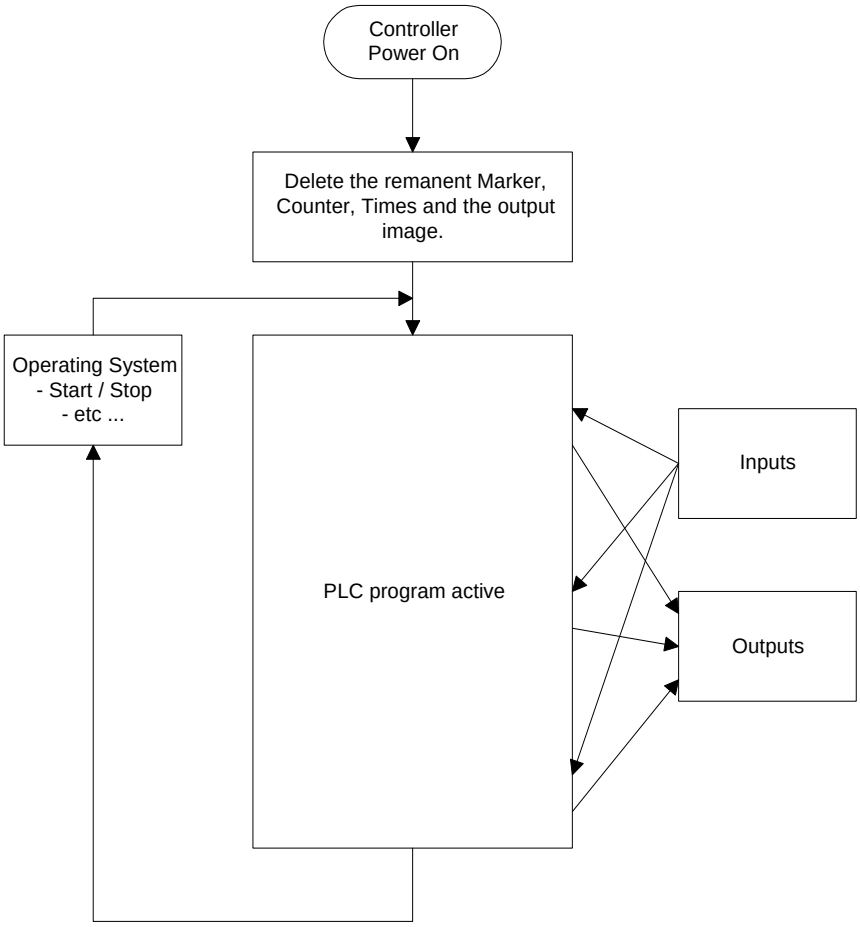



*Illustration 4-14: Principe de fonctionnement du système multitâches PacDrive*

PDM_UserMan_IEC_fr.fm

### 4.3.2 Types de données de la configuration de commande

Lors de la configuration d'un projet PacDrive™, les types de données sont automatiquement déclarés (voir aussi l'aide en ligne EPAS-4 PLC Configuration / Tableau : Objects of the PacDrive™ System).

Exemple :

Transmission d'une référence à un bloc.
Un MC-4 avec nom d'objet Axis_10 a été configuré.

```
Process(r_MC_4:=Axis_10);
```

Dans ce bloc Process, l'adresse logique peut être adressée de la manière suivante :

```
ControllerEnableSet(r_MC_4.logAdr);(* Validation pour
  le MC-4 *)
```

### 4.3.3 Taille du programme et des plages de variables

Voir également « Caractéristiques du système » à la page 231.

# 5 Recommandations pour la programmation

## 5.1 La notion de mappage

### 5.1.1 Structure des machines d’emballage

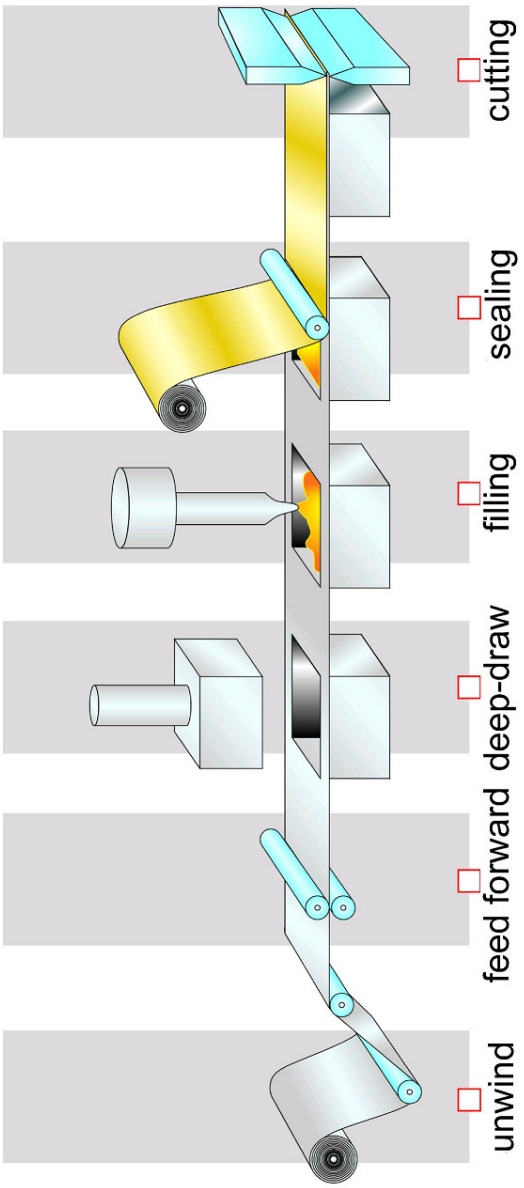



*Illustration 5-1: Exemple de la structure d’une machine d’emballage*

## 5.1.2 Le système d’automatisation

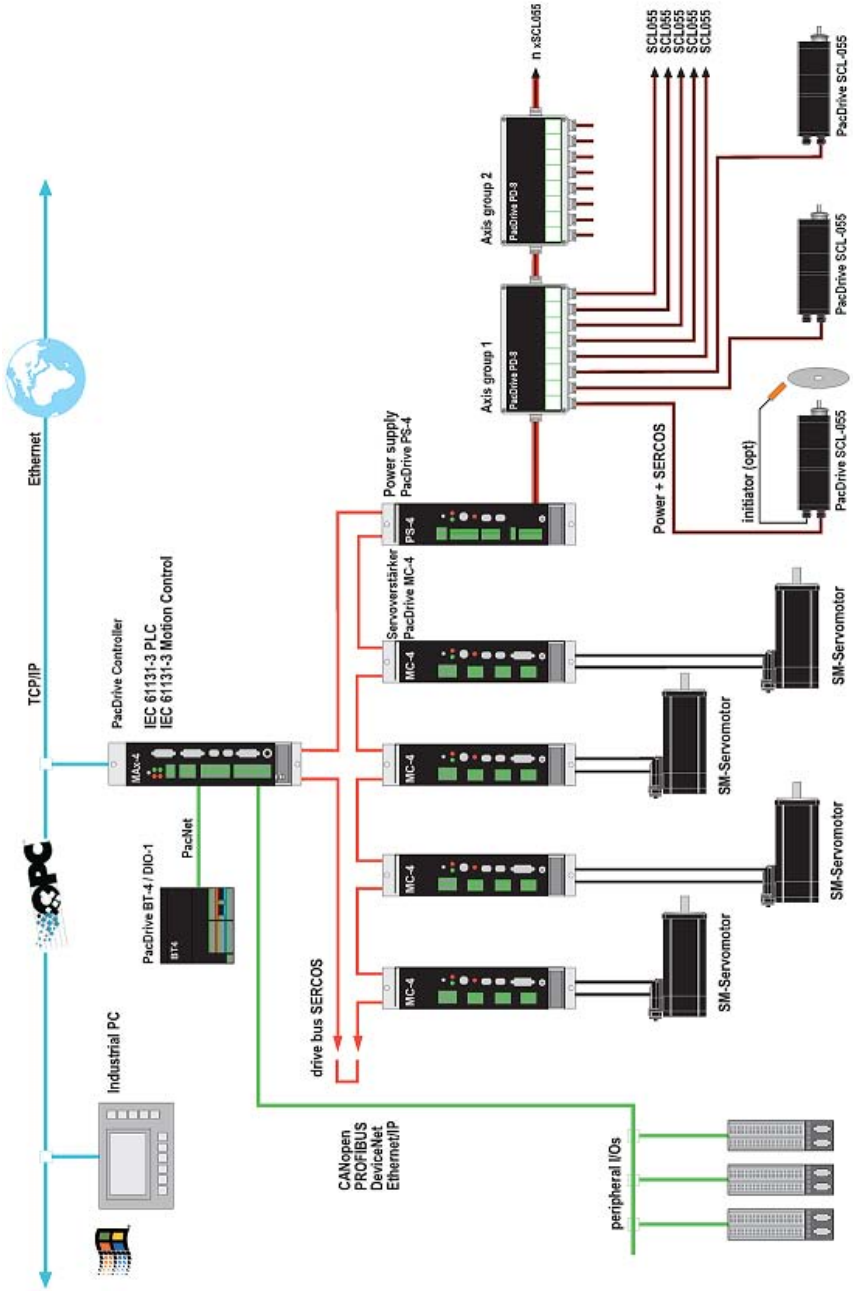



*Illustration 5-2: Vue d’ensemble du système d’automatisation PacDrive™*

PDM_UserMan_Richtlinien_fr.fm

### 5.1.3 De la définition du problème à la solution

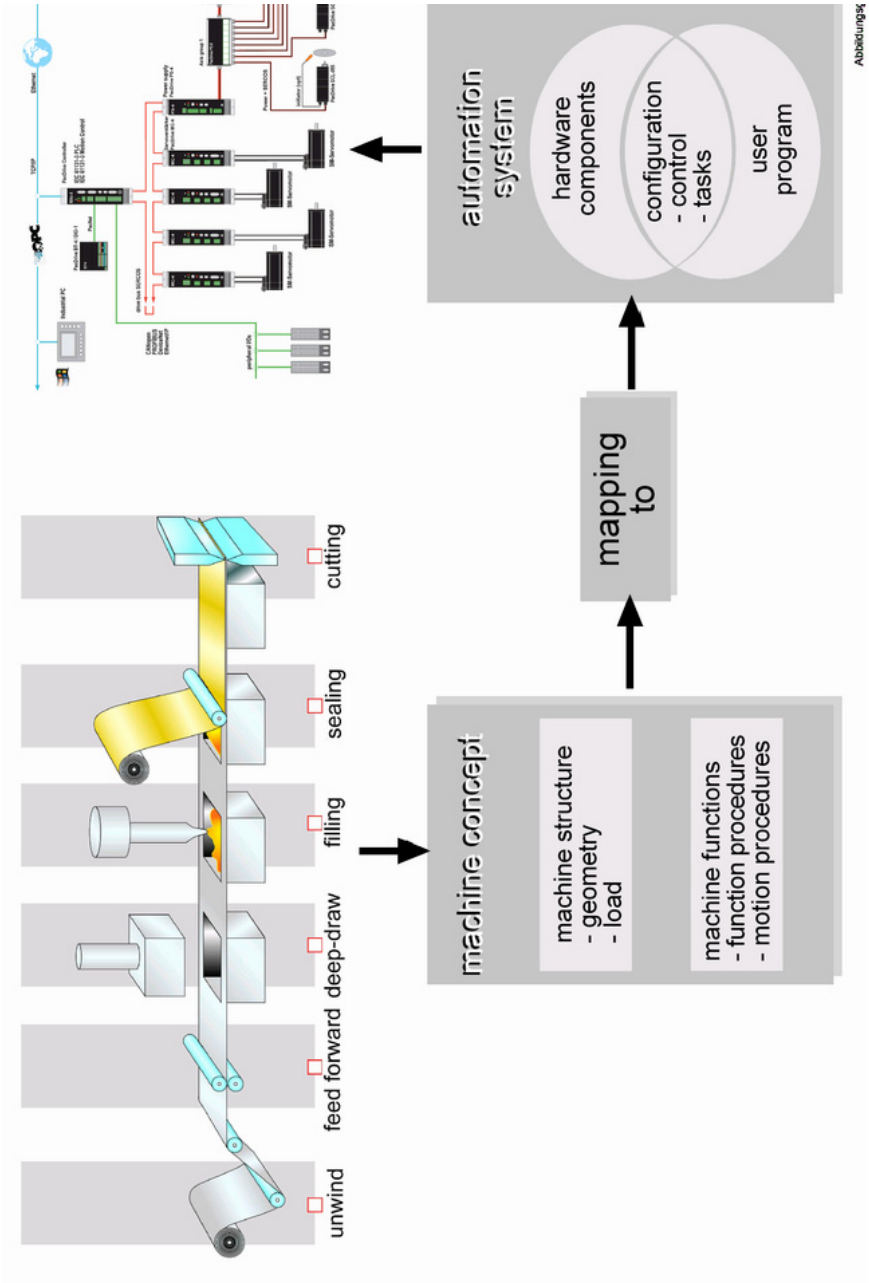



*Illustration 5-3: De la définition du problème à la solution avec PacDrive™*

## 5.1.4 Méthode de solution du problème

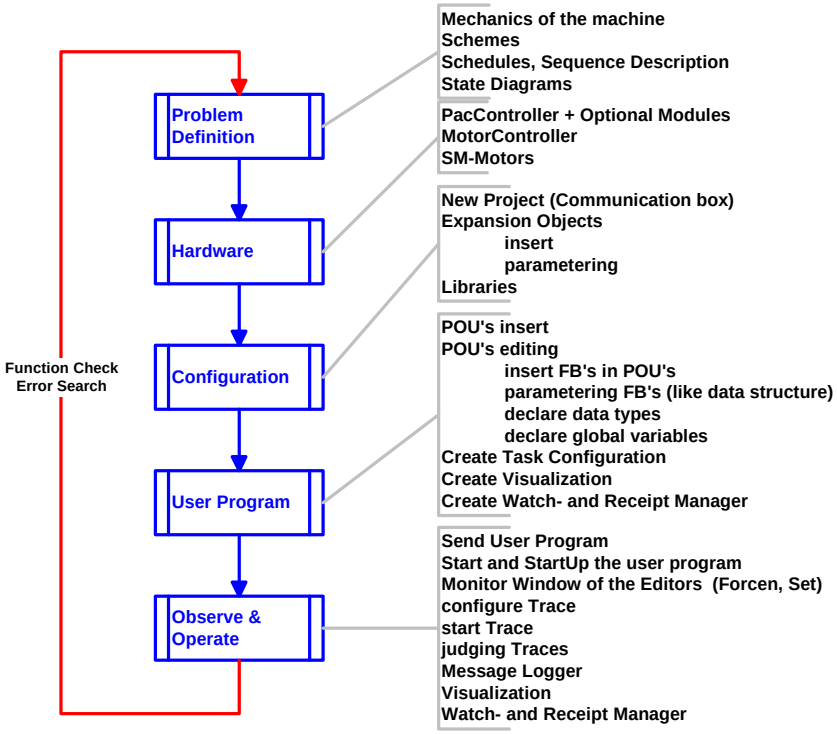



*Illustration 5-4: Méthode de solution du problème*

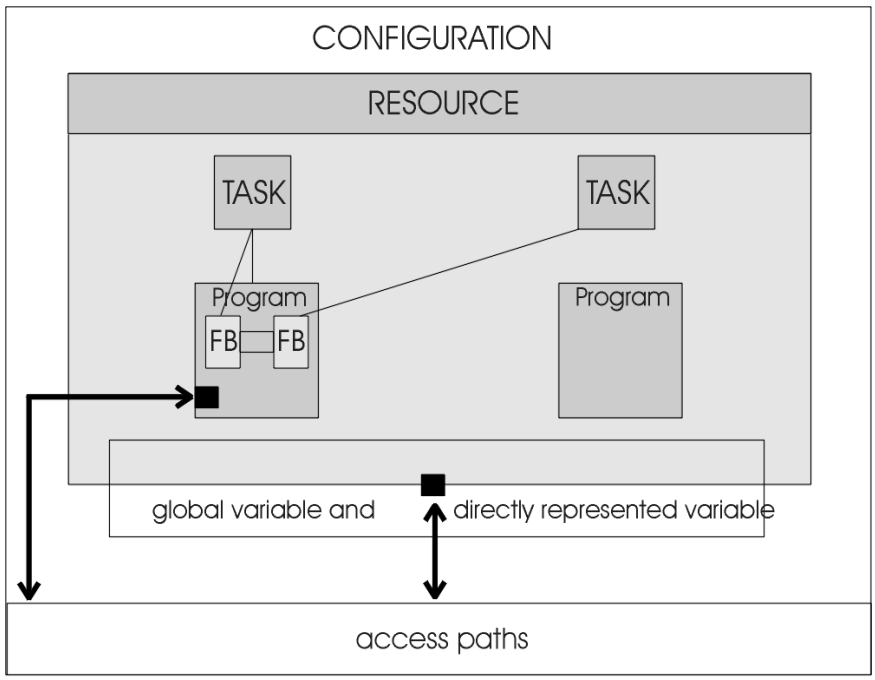
# 5.2 Structure d’un projet

## 5.2.1 Organisation d’un programme par tâches

**Généralités**

En règle générale, un programme PacDrive™ (programme de traitement) est composé de plusieurs tâches. Une tâche est un programme caractérisé par son importance (priorité) et son temps de cycle.

Une tâche permet de spécifier, par exemple, qu’une température ne sera contrôlée qu’à condition qu’aucune autre fonction plus importante ne soit en attente d’exécution. Ou bien elle spécifie qu’un contrôle ne peut avoir lieu moins de 30 secondes après le précédent.




Softwaremodell_englisch.cdr

*Illustration 5-5: Modèle logiciel*

Caractéristiques d’une tâche :

- elle se compose d’un ou de plusieurs blocs fonctionnels (UOP, unités d’organisation de programme)
- sa priorité va de 0 à 31 (0 étant la priorité la plus élevée)
- les tâches de priorité supérieure se substituent aux tâches d’un niveau de priorité inférieur
- les tâches de même niveau de priorité sont exécutées tour à tour par tranches de temps de 250 µs
- le nombre maximal de tâches gérées est de 127.

PDM_UserMan_Richtlinien_fr.fm

Le graphique suivant montre comment le système exécute les différentes tâches.

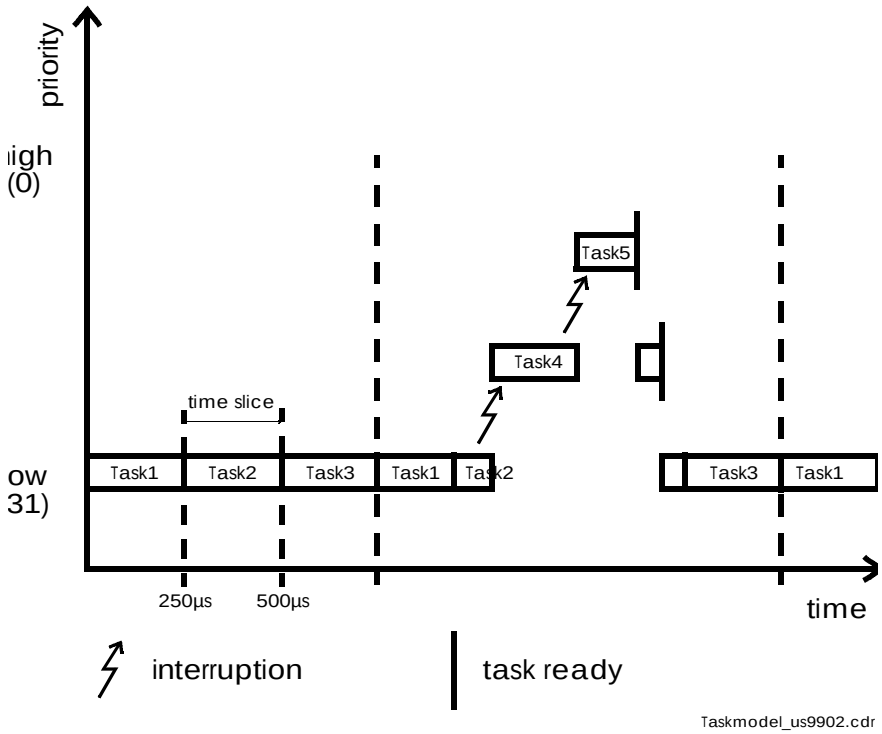



*Illustration 5-6: Modèle de tâches*

L'exécution des tâches à priorité inférieure est interrompue au profit de celle des tâches à priorité supérieure, et reprise ensuite.

Plusieurs tâches à priorité égale sont exécutées par tranches de temps, à moins qu'elles soient interrompues au profit de tâches à priorité supérieure. Cette procédure par tranches de temps prévoit que chaque tâche est exécutée pendant 250 µs avant d'être mise en attente pour l'exécution d'une autre tâche du même niveau de priorité.

Page suivante : exemple pratique d'un traitement par tâches.

1) passage à une tâche à priorité supérieure

2) passage à une tâche à priorité égale à la fin d'une tranche de temps

3) passage à une tâche à priorité supérieure

PDM_UserMan_Richtlinien_fr.fm

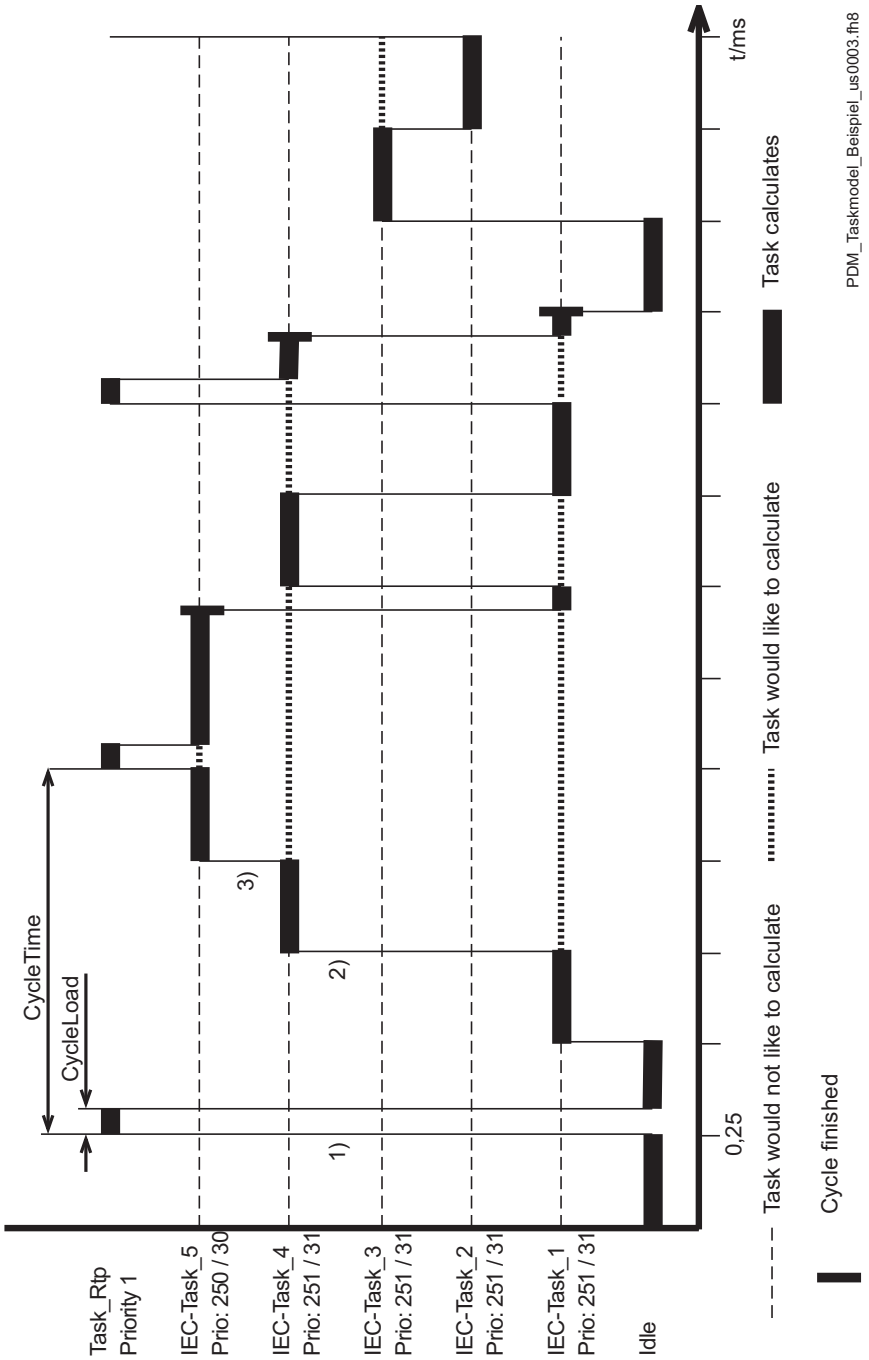
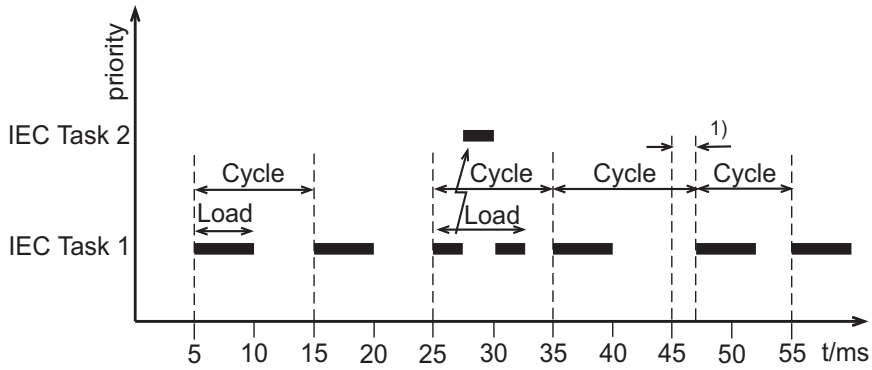



*Illustration 5-7: Exemple pratique d'un modèle à tâches*

Ce graphique décrit le rapport entre la durée prévue de l'intervalle réservée à la tâche, la durée réelle de l'intervalle et la durée de charge. Valeurs spécifiées pour le graphique :

- Durée prévue de l'intervalle : 10ms
- Cycle correspond à la durée réelle de l'intervalle
- Load est la durée d'exécution ou de charge de la tâche




1): This delay can be caused e.g. by a task with higher priority or by the system.

Note: This is a greatly simplified representation of the real behaviour.

PDM_CycleLoad_us9912.fh8

*Illustration 5-8: Exemple du comportement de Cycle et Load*

Le système essaie de respecter la durée prévue de l'intervalle. Mais la durée réelle de l'intervalle peut être supérieure à la durée prévue. C'est notamment le cas quand le démarrage d'un cycle est retardé par le système ou une tâche à priorité supérieure. Le système essaie alors de raccourcir le cycle suivant (Cycle) pour rentrer dans la grille de temps.

L'exécution des programmes associés à une tâche est signalée par Load. Si l'exécution des programmes ne s'achève pas avant la fin du temps prévu pour l'intervalle, le système transmet un message de diagnostic (cycle time transgression [dépassement de la durée de cycle] ou serious cycle time transgression [dépassement important de la durée de cycle]).

**REMARQUE**

Plusieurs tâches système sont exécutées selon les mêmes principes. Ces tâches système ainsi que la classification de la priorité IEC se trouvent sous Quelle priorité est la bonne ?

**Développement d’un modèle de tâches spécifique au projet**

Le développement d’un modèle de tâches est indispensable au début d’un projet. Il s’agit de définir les éléments suivants :

- nombre des tâches
- priorité des tâches
- temps de cycle

**REMARQUE**

Les paramètres lus ou écrits via SERCOS risquent de ralentir considérablement l’exécution des programmes (p.ex., TRACE, Error Quit, ...).
cf. l’aide en ligne EPAS-4 / PLC Configuration / Types of Object Parameters

De combien de tâches ai-je besoin ?

Une seule tâche est très souvent suffisante pour la fonction de mouvement, voire même le projet intégral. Des tâches supplémentaires seront éventuellement nécessaires pour les fonctions de surveillance, les fonctionnalités API, les appareils de commande opérateur, etc. Le nombre de tâches doit être aussi réduit que possible.

Remarque de principe :

- Autant que faire se peut, le nombre de tâches doit être réduit à une seule.
- Les fonctions dont l’exécution nécessiterait un temps plus long (p.ex., services de données, accès aux paramètres SERCOS) doivent être associées à une tâche à priorité inférieure.
- Une tâche à priorité supérieure peut en revanche se révéler nécessaire, par exemple, pour tel ou tel aspect de la sécurité.
- Un nombre de tâches inférieur ou égal à trois est en règle générale suffisant.
- Les tâches à priorité égale sont à éviter. Il est préférable de faire appel aux fonctions du « programme de lancement ». Cette solution assure un ordonnancement temporel défini des tâches.

Quelle priorité est la bonne ?

Le système PacDrive™ connaît 252 priorités internes. Les 32 priorités les plus basses sont disponibles pour les programmes IEC. Le graphique ci-dessous permet de situer les priorités IEC par rapport aux priorités système.

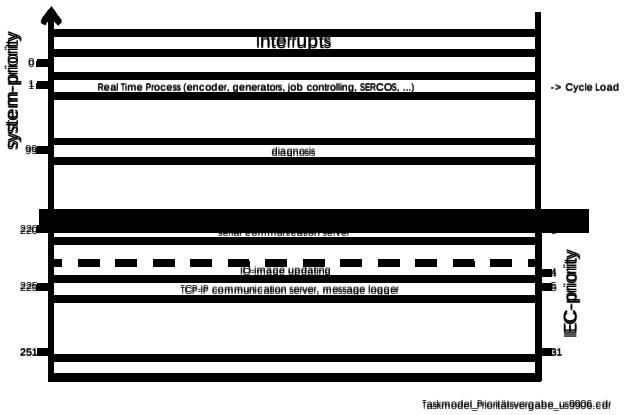



*Illustration 5-9: Modèle de tâches / attribution des priorités*

Règle de base pour les programmes utilisateurs :

- Lorsque le programme utilisateur ne comporte qu'une seule tâche la priorité attribuée n'a que peu d'importance. Il convient de choisir une priorité entre 6 et 31.
- Il est utile d'associer les fonctions dont l'exécution nécessite un temps relativement long (services de fichiers, accès aux paramètres SERCOS) ou dont l'exécution peut être décalée (régulateur de chauffage) à une tâche à priorité inférieure.
- Une tâche à priorité supérieure peut s'avérer nécessaire pour les aspects de sécurité.
- Si le temps de cycle d'une tâche ne doit pas dépasser 5 ms, sa priorité doit être inférieure à 5 pour éviter des interruptions par le serveur de communication TCP/IP.

**REMARQUE**

D’éventuelles erreurs de cycle sporadiques (505 « simple cycle error [erreur de cycle simple] », 506 « multiple cycle error [erreur de cycle multiple] ») peuvent être dues au ralentissement de l’exécution de vos tâches IEC par les procédures de communication Ethernet (Ftp, Ping, EPAS-4, …).
Nous conseillons d’attribuer à ce type de tâches une priorité de tâche IEC supérieure à 5 (priorité de 0 à 4).
Si la priorité d’une tâche IEC utilisant les modules d’un bus de terrain est supérieure (0 à 3) ou égale (4) à celle de l’actualisation de la table E/S, la cohérence des données n’est plus assurée que pour l’accès à un seul type d’élément (octet, booléen, mot …).

Quel temps de cycle choisir ?

Pour choisir le temps de cycle, il convient de tenir compte des éléments suivants :

- temps de cycle du mouvement
- temps minimal d’exécution d’une fonction FAO
- accès à un paramètre SERCOS (retard! -> cf. aide en ligne EPAS-4 / PLC Configuration)
- exécution de fonctions avec retard.

**REMARQUE**

La durée typique d’un cycle se situe entre 5 et 10 ms.
Une durée de moins de 5 ms peut être obtenue éventuellement par modification de la priorité de la tâche.
-> cf. Quelle priorité est la bonne ?

**Surveillances**

Types de surveillance

L'exécution des tâches cycliques est surveillée. Messages de diagnostic possibles :

| Classe (valeur par défaut) | Code de diagnostic | Texte de diagnostic spécifique à la langue | Acquittable (valeur par défaut) |
|---|---|---|---|
| 4 | 313 | serious cycle time transgression [dépassement important du temps de cycle] | non |
| 6 | 317 | cycle time transgression [dépassement du temps de cycle] | oui |

Causes et réparations possibles

| **313 serious cycle time transgression [dépassement important du temps de cycle]** | |
|---|---|
| Un dépassement important du temps de cycle réservé à une tâche IEC est signalé lorsque le temps mesuré dépasse 10 temps de cycle (cf. configuration de tâches). La tâche passe à l'état « erreur ». Cet état ne peut être quitté que par une remise à zéro de la tâche.<br>Remarque La supervision des temps de cycle définis pour les tâches IEC utilise le paramètre d'objet Load représenté dans l'objet d'extension « tâche IEC ». | |
| Cause 1 :<br><br>Solution : | Exécution d'un paramètre ou d'un bloc fonctionnel gourmand en temps d'exécution.<br>Vérifier le programme (p.ex. l'accès aux paramètres SERCOS -> cf. types de paramètres d'objet ou fonction WAITT). |
| Cause 2 :<br>Solution : | Intervalle de temps trop court.<br>Vérifier la configuration de la tâche. |

| **317 cycle time transgression [dépassement du temps de cycle]** | |
|---|---|
| Un « léger » dépassement de cycle vient d'être signalé. Cette signalisation intervient en cas de dépassement du temps de cycle de plus de 250 µs. Le temps de cycle actuel, respectivement le temps de cycle spécifié dans la configuration de tâches sont indiqués en ms dans DiagExtCode. | |
| Cause :<br>Solution : | cf. message de diagnostic 313 |

PDM_UserMan_Richtlinien_fr.fm

### Le concept FastTask

Le concept FastTask permet de programmer des tâches IEC rapides et synchrones à priorité élevée et commandées par interruption.

Pour plus d'informations sur le concept FastTask, cf. l'aide en ligne EPAS-4 >library MAx-4 > IEC_Tasks > General Notes on the FastTask concept.

**REMARQUE**

Avant tout recours au concept FastTask, nous vous recommandons de l'étudier à fond et de prendre contact avec notre service d'application.

### La machine d'état des tâches IEC

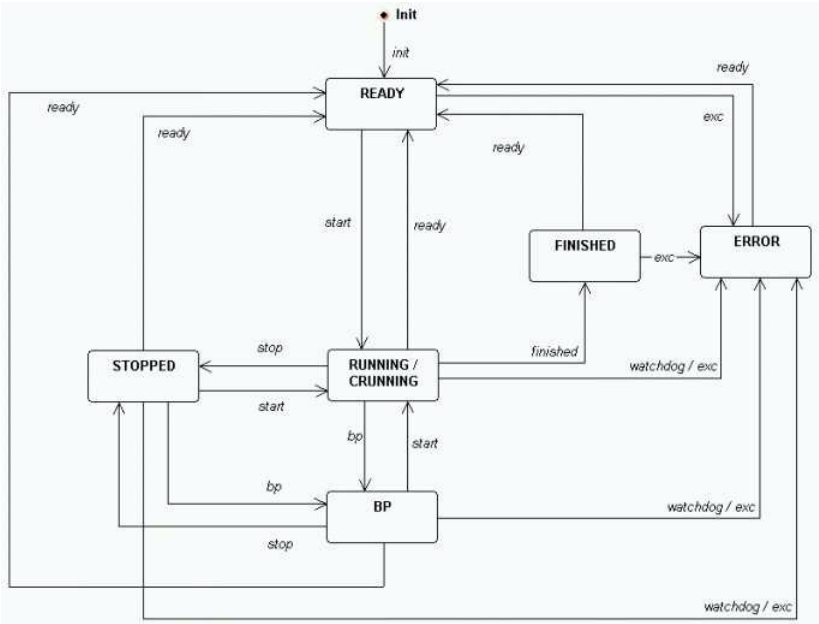



*Illustration 5-10: Machine d'état des tâches IEC*

L'état de la tâche peut être lu à l'aide de l'objet de configuration « tâche IEC » (état de paramètre).

| Valeur | Désignation | Signification |
|---|---|---|
| 0 | | pas de tâche<br>état d'initialisation, pas de tâche définie. |
| 1 | READY | prête<br>tâche définie et prête à démarrer. |
| 2 | RUNNING | en cours d'exécution<br>tâche en cours d'exécution. (versions < 00.10.00 seulement) |

| Valeur | Désignation | Signification |
|---|---|---|
| 3 | CRUNNING | exécution cyclique<br>exécution cyclique de la tâche. |
| 4 | STOPPED | arrêt<br>arrêt de la tâche. |
| 5 | BP | arrêt à Break Point<br>arrêt de la tâche par un point d’arrêt ou en mode intermittent. |
| 6 | ERROR | erreur<br>détection d’une erreur grave ; l’exécution de la tâche est suspendue. |
| 7 | FINISHED | terminée<br>l’exécution de la tâche est terminée. |

*Tableau 5-1: Valeurs de la machine d’état des tâches IEC*

PDM_UserMan_Richtlinien_fr.fm

**Possibilités de mesure**

Il est possible de mesurer la charge des tâches IEC. Pour ce faire, intégrez l'objet « tâche IEC » dans la configuration de commande. Description des paramètres, cf. aide en ligne EPAS-4 / PLC Configuration.

Marche à suivre :

- Dans la configuration de commande, intégrez un objet de type tâche IEC pour chaque tâche programmée dans la configuration de tâches.



**REMARQUE**

Le premier objet de type tâche IEC de la configuration de commande est associé à la première tâche de la configuration de tâches. Les n premiers objets de type tâche IEC seront associés aux n premières tâches de la configuration de tâches. Le système peut en principe comporter plus ou moins d'objets de type tâche IEC que de tâches.

- Transmettez l'objet au PacDrive Controller Max-4.
- Pour commencer la mesure, mettez EnableLoad sur / TRUE.

## 5.3 Structure du programme ELAU

Pour structurer efficacement votre programme, vous devez notamment savoir dans quelles parties les blocs fonctionnels (UOP = Unités d'Organisation de Programme) diffèrent les unes des autres. Voir également « Bases de la norme IEC 61131 » à la page 25.

Réalisez le niveau le plus élevé de votre programme en langage SFC (diagramme fonctionnel en séquence).

PDM_UserMan_Richtlinien_fr.fm

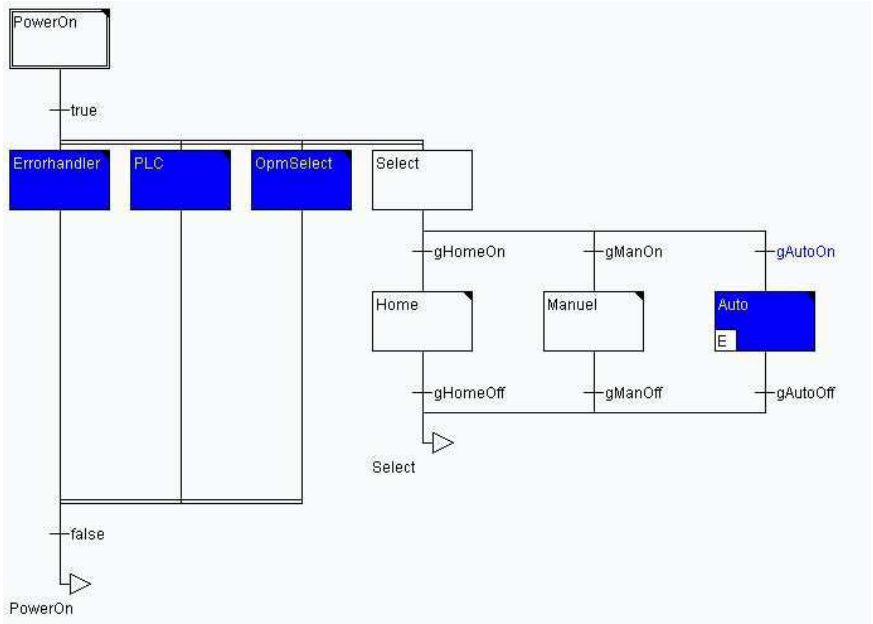



*Illustration 5-11: Exemple d'un niveau de programme « supérieur » en SFC*

Les parties de programme à niveau inférieur doivent être réalisés en FBD.

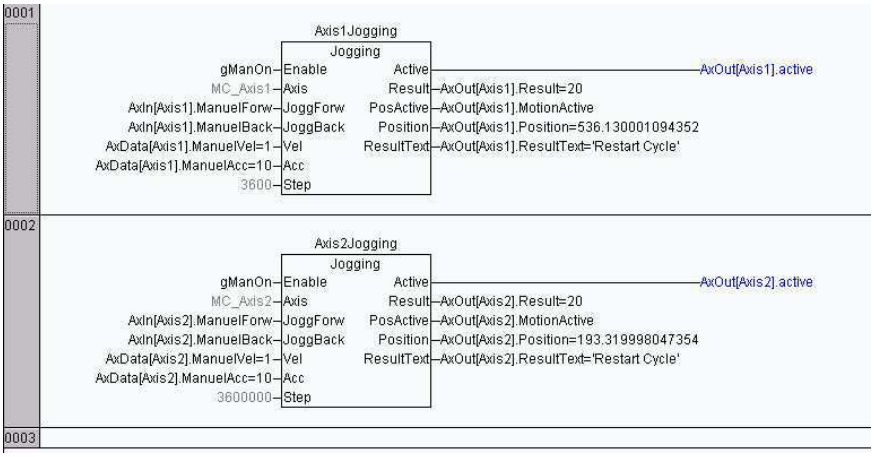



*Illustration 5-12: Exemple d'un programme de niveau inférieur en FBD*

Le langage ST se prête particulièrement bien aux fonctions et blocs fonctionnels plus complexes puisqu'il offre des instructions confortables pour la structuration (p.ex. WHILE, FOR, CASE, …).

PDM_UserMan_Richtlinien_fr.fm

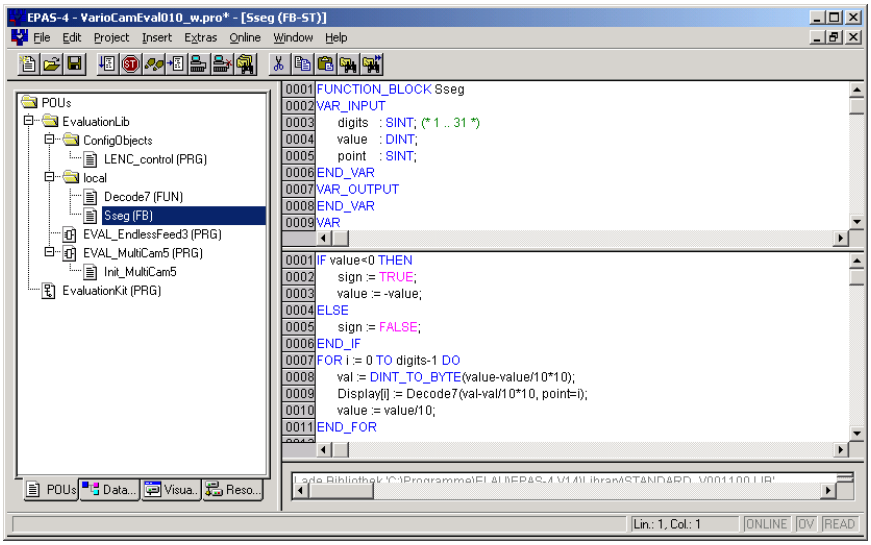



*Illustration 5-13: Exemple d'un bloc fonctionnel en ST*

### L'utilisation de machines d'état

La pratique de la programmation a largement démontré l'utilité – et très souvent aussi la nécessité – d'adosser la réalisation de programmes, fonctions et blocs fonctionnels à des machines d'état.

Les machines d'état sont programmées en littéral structuré (ST) à l'aide de l'instruction CASE.

```
CASE lState OF
    1:    (* initialisation *)
          < Instructions >
          lState:=lState+1;(* passer à l'état suivant *)

    2:    (* état 2 *)
          < Instructions >
          lState:=lState+1;(* passer à l'état suivant *)
    :
    n:    (* état n *)
          < Instructions >
          lState:=x;    (* passer à l'état suivant *)
END_CASE;
```

PDM_UserMan_Richtlinien_fr.fm

### 5.3.1 Fonctions et blocs fonctionnels d’ELAU

ELAU propose un nombre important de bibliothèques comportant des fonctions et des blocs fonctionnels pour certains domaines stratégiques.

- Standard.lib
  Bibliothèque IEC standard comportant des fonctions élémentaires (p.ex. compteurs, fonctions de cordons de caractères, temporisateurs etc.).
- MAx-4.lib
  Bibliothèque des interfaces pour les fonctions de base du PacDrive Controller Max-4.
- Check.lib
  Bibliothèque des interfaces pour la vérification de variables ARRAY (fonction de mise au point et de test).
- Basic.lib
  Bibliothèque IEC de fonctions de mouvement. Bibliothèque constituée à partir de la MAx-4.lib.
- IECsfc.lib
  En SFC, utilisation de pas IEC depuis V00.06.00.
- CSpline.lib
  Bibliothèque des interfaces pour les fonctions VarioCam® CSpline permettant de convertir en ligne les données de point de référence en données de graphique.
- TorqueCam.lib
  Bibliothèque des interfaces pour les cames à couple.
- CANL2.lib
  Bibliothèque des interfaces pour CAN Layer 2.
- CANopen_M.lib
  Bibliothèque des interfaces pour CANopen Master
- CANopen_S.lib
  Bibliothèque des interfaces pour CANopen Slave.
- PBDP_M.lib
  Bibliothèque des interfaces pour PROFIBUS-DP Master.
- PBDP_S.lib
  Bibliothèque des interfaces pour PROFIBUS-DP Slave.
- DNet_S.lib
  Bibliothèque des interfaces pour Device Net Slave
- HMI_pcs.lib
  Bibliothèque des interfaces pour le couplage à Lauer PCS
- HMI_Simatic.lib
  Bibliothèque des interfaces pour le couplage à Simatic.
- ModBus.lib
  Bibliothèque des interfaces pour le couplage à ModBus.
- TSC.lib
  Bibliothèque des interfaces pour le couplage à ModBus (cf. Guide d’utilisation HMI pour les bibliothèques d’interfaces homme-machine).

PDM_UserMan_Richtlinien_fr.fm

- VarioCam.lib
  Bibliothèque des fonctions de mouvement complexes.
- Bibliothèque PLC
  Fonctions binaires, communication, régulateurs PID, fonctions de conversion, etc.

**Autres bibliothèques (produits tiers)**

- Bibliothèque de robotique
  Bibliothèque pour des applications dans le domaine de la robotique

**Caractéristiques de traitement**

Les temps d'exécution des fonctions et blocs fonctionnels diffèrent énormément. Quelques exemples à titre d'illustration :

- MultiCam3(): durée typique : 12 µs
- AppendToWriteFile(): quelques centaines de ms

L'utilisation de blocs fonctionnels à durée d'exécution prolongée suppose souvent une augmentation temporaire de la surveillance de cycle définie pour la tâche. La fonction *CycleCheckTimeSet()* permet de le faire.

## 5.3.2 Transmission des paramètres

Remarques relatives à la transmission des paramètres :

**VAR_IN / VAR_OUT**

Transmission de la valeur : la copie de la valeur est en cours. Cette opération nécessite de la capacité mémoire et du temps de calcul. C'est pourquoi il s'agit d'éviter la transmission de variables longues (ARRAY et STRUCT).



**REMARQUE**

Les types de données de la configuration de commande (MC_4, D_IN, ...) sont transmises de façon automatique en tant que référence.
En cas de transmission comme référence, les variables sont transmises dans le programme de façon apparemment « normale ». Mais sur le plan interne, ce n'est pas une variable, mais seulement un pointeur qui est transmis.

**VAR_IN_OUT**

Transmission de l'adresse de la variable.
Ce choix présente l'avantage de ne pas déclencher une opération de copie et permet donc d'économiser des ressources et du temps de calcul.

## 5.4 Identificateurs

Les conventions de programmation suivantes assureront une identification homogène des objets, variables et instances d'objets et de variables. Cette démarche simplifie la lecture et la compréhension du code, et notamment des interfaces des fonctions et des blocs fonctionnels. Elle facilite ainsi la maintenance du code.

Ces conventions n'étant pas prévues dans la norme IEC 61131-3, elles sont à considérer comme une aide aux utilisateurs et programmeurs de PacDrive™.

Les identificateurs doivent comporter des préfixes homogènes permettant de simplifier l'identification du type d'objet.

| Préfixe | Type | Exemple |
|---|---|---|
| TASK_ | Task | TASK_Main |
| FB_ | Bloc fonctionnel | FB_Pos |
| ST_ | Structure | ST_BufferEntry |
| E_ | Type de donnée énuméré (ENUM) | E_PosStates |
| T_ | Référence (TYPE) | T_Nibble |
| P_ | Programme | P_Main |
| F_ | Fonction | F_Convert |

*Tableau 5-2: Définition de noms d'objet*

| Préfixe | Type | Exemple |
|---|---|---|
| fb | Nom d'instance d'un bloc fonctionnel | fbPos |
| st | Nom d'instance d'une structure | stBufferEntry |
| e | Nom d'instance d'un type de donnée énuméré (ENUM) | ePosStates |
| aucun | Nom d'instance d'un type de référence | Nibble |

*Tableau 5-3: Génération des instances d'objets*

Identifiez vos variables par des noms « parlants ». Nous recommandons deux types de préfixe :

- Préfixe de portée
- Préfixe de type

| Préfixe de portée | Type | Exemple |
|---|---|---|
| pas de préfixe | Variable locale | bName |

PDM_UserMan_Richtlinien_fr.fm

| Préfixe de portée | Type | Exemple |
|---|---|---|
| I_ | Entrée | I_bSleighThere |
| O_ | Sortie | O_bValveOpen |
| g_ | Variable globale | g_bName |
| r_ | Référence (p.ex. à un objet MC_4) | r_MC_4 |

*Tableau 5-4: Préfixe de portée*

| Préfixe de type | Type | Mots clés (Bit) | Exemple |
|---|---|---|---|
| b | BOOL | booléen (1) | bName |
| c | BYTE | Cordon de longueur 8 bits (8) | cName |
| w | WORD | Cordon de longueur 16 bits (16) | wName |
| lw | DWORD | Cordon de longueur 32 bits (32) | lwName |
| si | SINT | Entier court (8) | siName |
| i | INT | Entier (16) | iName |
| l | DINT | Entier double (32) | lName |
| usi | USINT | Entier court (8) | usiName |
| ui | UINT | Entier (16) | uiName |
| ul | UDINT | Entier double (32) | ulName |
| r | REAL | Nombre réel (32) | rName |
| lr | LREAL | Réel long (64) | lrName |
| d | DATE | Date (32) | dName |
| to | TOD | Heure du jour (32) | toName |
| dt | DT | Date et heure du jour (32) | dtName |
| t | TIME | Durée (32) | tName |
| s | STRING | Cordon de caractères de longueur variable | sName |
| p | Pointer | Pointeur | pbName |
| a | Array | Tableau | abName |

*Tableau 5-5: Préfixe de type*

## 5.5 Majuscules/minuscules

Les noms d'objet comportent toujours un préfixe en majuscules. Le préfixe et le nom d'objet proprement dit sont séparés par le trait de soulignement « _ ». Les noms de variable et les noms d'instance comportent toujours un préfixe en minuscules. Un identificateur commence toujours par une majuscule. Chaque mot d'un identificateur composé de plusieurs mots commence par une majuscule. Un séparateur entre deux mots (p.ex. _) n'est pas recommandé.



**REMARQUE**

EPAS-4 ne distingue pas les minuscules et les majuscules.

## 5.6 Signes autorisés

Chiffres, caractères et caractères spéciaux autorisés dans les identificateurs :

0 ... 9, A ... Z, a ... z



**REMARQUE**

Il est recommandé de choisir les noms en langue anglaise exclusivement.

## 5.7 Réalisation d'un projet PacDrive™

### 5.7.1 Définition du problème

Nous allons réaliser un projet simple. Le projet est défini à travers une seule contrainte : la sortie de MAx-4 doit clignoter à une fréquence de 1 Hz.

PDM_UserMan_Richtlinien_fr.fm

### Définition des valeurs d’entrée et de sortie, respectivement des senseurs et des actuateurs

| Equipement | Nom d’entrée/sortie | Commentaire |
|---|---|---|
| H1 | O_bEquipmentOn-IndicatorH1 | Un voyant connecté à la sortie clignotera à une fréquence de 1 Hz tant que l’installation est en marche. |

*Tableau 5-6: Définition E/S*



**REMARQUE**

Une description ambiguë peut être à l’origine d’une mauvaise interprétation de la fonction à réaliser. Veuillez donc apporter le plus grand soin à la rédaction de vos descriptifs. Si nécessaire, complétez votre descriptif par des graphiques.

| Lettre d’identification | Type d’équipement | Exemples |
|---|---|---|
| A | Sous-ensembles | |
| B | Convertisseur de grandeurs non électriques en grandeurs électriques et inversement | Convertisseurs thermiques, de lumière, de fréquence de rotation, etc. ; détecteurs de proximité, convertisseurs d’angle et de course. |
| C | Condensateurs | |
| D | Eléments binaires, temporisations et mémoires | |
| E | Divers | |
| F | Protections | |
| G | Alimentations, générateurs | |
| H | Avertisseurs | |
| K | Contacteurs, relais | |
| L | Inductivités | |
| M | Moteurs | |
| N | Amplificateurs, régulateurs | |
| P | Mesures et contrôles | |
| Q | Interrupteurs de puissance | |

| Lettre d’identification | Type d’équipement | Exemples |
|---|---|---|
| S | Commutateurs, sélecteurs | Boutons-poussoirs, interrupteurs-limiteurs, auxiliaires de commande, cadrans d’appel |
| T | Transformateurs | |
| U | Modulateurs, convertisseurs entre grandeurs électriques | |
| V | Semiconducteurs, tubes | |
| W | Bus, câbles, antennes | |
| X | Bornes, fiches mâles et femelles | |
| Y | Installation mécanique à commande électrique | Freins, embrayages, soupapes |
| Z | Filtres, correcteurs, limiteurs, terminaisons | |

*Tableau 5-7: Identification des équipements électriques*

### Représentation du problème sur l’axe temps

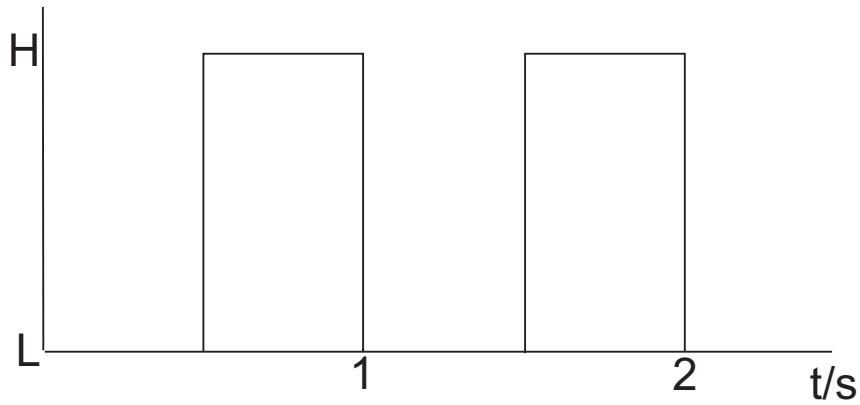



*Illustration 5-14: Représentation sur l’axe temps*

## 5.7.2 Matériel

Un seul PacDrive Controller MAx-4 suffit pour résoudre ce problème.

PDM_UserMan_Richtlinien_fr.fm

## 5.7.3 Configuration

### Créer un projet

- Cliquez sur FILE | NEW pour créer un nouveau projet.

Une boîte de dialogue s’affiche dans laquelle vous devez spécifier les paramètres de la connexion en ligne.
(-> pour plus d’informations, voir le Mode d’emploi EPAS-4)

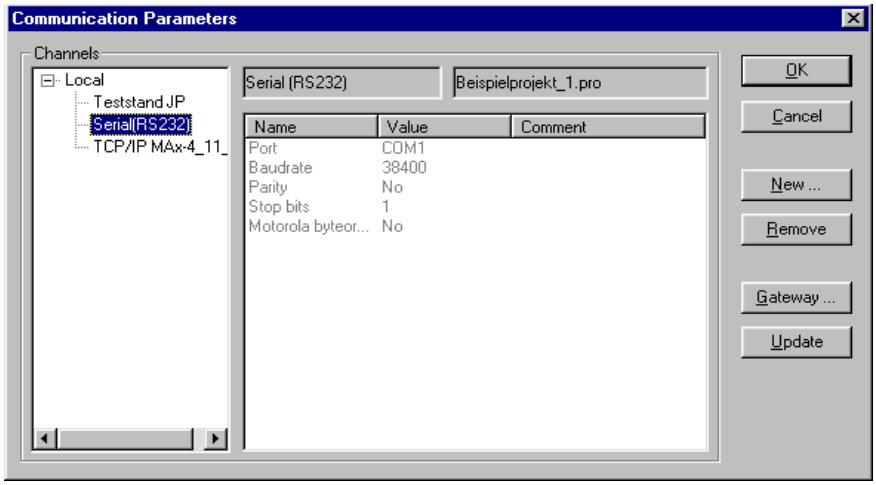



*Illustration 5-15: Boîte de dialogue « Communication Parameters » (Paramètres de communication)*

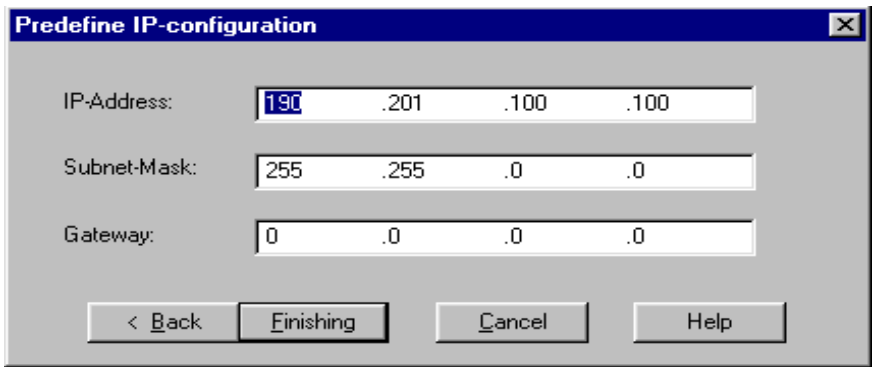



*Illustration 5-16: Assistant d’adressage IP*

### Configuration du projet

Les éléments constitutifs du système PacDrive™ ainsi que leurs caractéristiques sont définies dans la configuration de commande.

PDM_UserMan_Richtlinien_fr.fm

Un système PacDrive™ est composé d'au moins un PacDrive Controller. Ce PacDrive Controller permet en principe de disposer des groupes de configuration suivants.

| Groupe | Description |
|---|---|
| Données techniques générales | Paramètres généraux du PacDrive Controller |
| Diagnostic | Paramètres de diagnostic du PacDrive Controller |
| Versions | Paramètres de version du PacDrive Controller |
| Bus en temps réel | Objet du PacDrive Controller |
| Groupe de sorties numériques | Objet du PacDrive Controller |
| Groupe d'entrées numériques | Objet du PacDrive Controller |
| Groupe d'entrées de mesure numériques | Objet du PacDrive Controller |

*Tableau 5-8: Groupes / objets de configuration de base*

- Sélectionnez *Resources* pour appeler l'éditeur de configuration.

*Illustration 5-17: Sélection de ressources dans l'organiseur d'objets*

- Sélectionnez maintenant la fonction *PLC Configuration* (Configuration de commande).

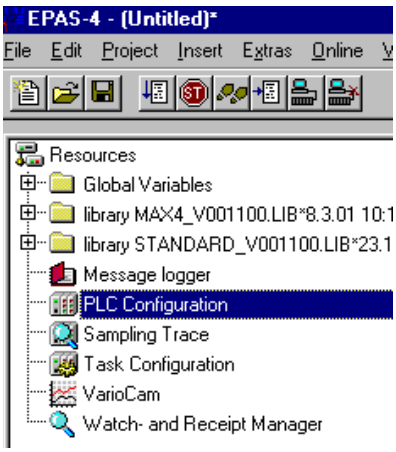



*Illustration 5-18: Sélection de la configuration de commande*

L'écran propose la configuration minimale d'un système PacDrive™. Cette configuration suffira pour réaliser notre premier exemple.

PDM_UserMan_Richtlinien_fr.fm

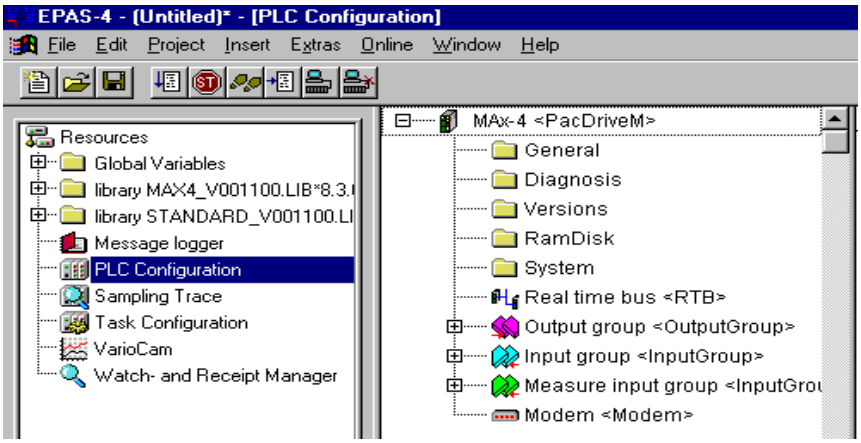



*Illustration 5-19: Configuration de commande avec configuration minimale*

Deux fichiers pour la configuration de commande sont stockés dans la mémoire du PacDrive Controller : le *fichier de configuration* et le *fichier de paramétrage*.

La *Configuration de commande* gère en principe les objets (MotorController MC-4, codeurs logiques, etc.).
Le *fichier de paramétrage* contient les « entrées avec initialisation d'utilisateur » (cf. Types d'objets du PacDrive Controller MAx-4) avec leurs valeurs d'initialisation.

**REMARQUE**

En cas de modification du *fichier de configuration de commande* (objet ajouté/supprimé), le PacDrive Controller doit être remis à zéro (ou réamorcé).
Pour plus d'informations, voir l'aide en ligne EPAS-4.

Notre projet ne nécessite en principe aucune modification de la configuration par défaut. Mais nous allons donner un nom symbolique « parlant » à notre sortie. Ce sera le nom défini dans l'exposé du problème.

- Entrez ce nom en cliquant sur le nom symbolique entre parenthèses après output_0.

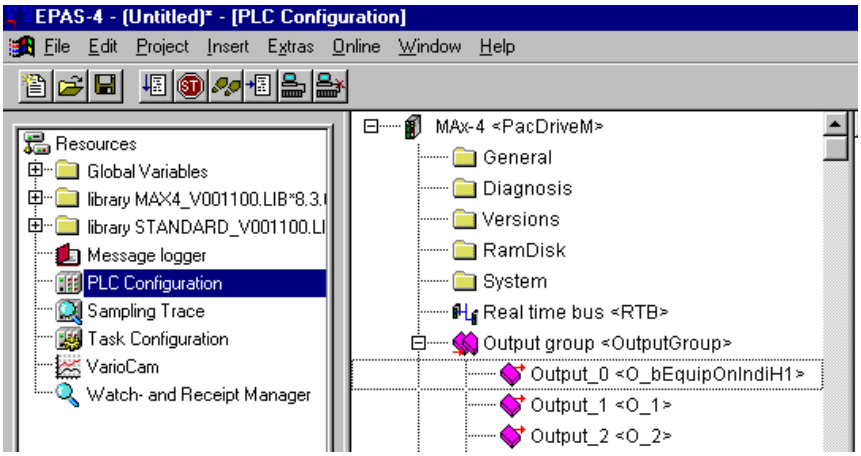



*Illustration 5-20: Attribution d'un nom symbolique à la sortie « output_0 »*

## 5.7.4 Programme utilisateur

- Passez aux blocs fonctionnels pour entrer le programme.

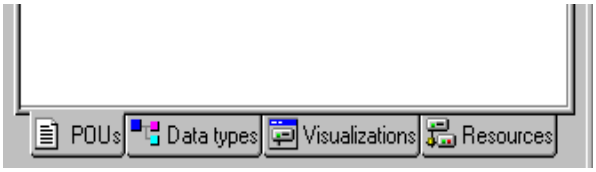



*Illustration 5-21: Sélection de blocs fonctionnels dans l'organiseur d'objets*

- Cliquez à droite sur « POUs » et sélectionnez « Add object … ».

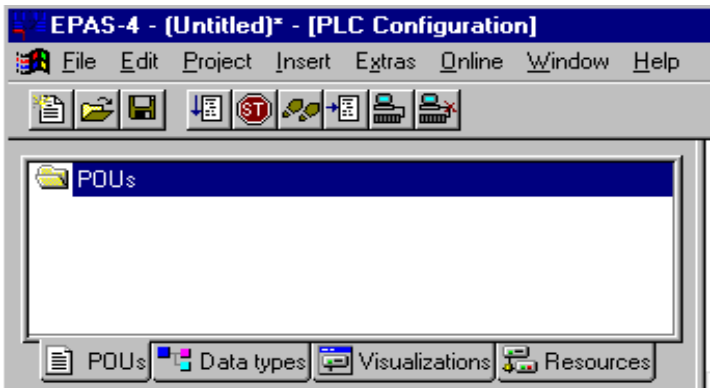



*Illustration 5-22: Sélectionnez les blocs fonctionnels en cliquant à droite.*

- Identifiez votre programme par un nom « parlant » et confirmez par OK.

PDM_UserMan_Richtlinien_fr.fm

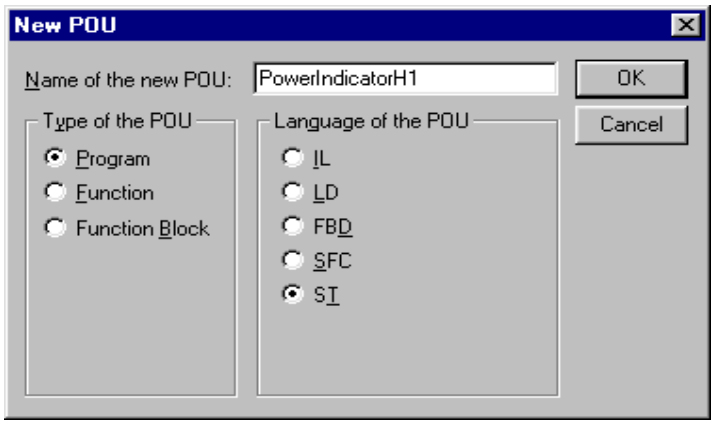



*Illustration 5-23: Boîte de dialogue « New POU » pour l’insertion d’un bloc fonctionnel*

Vous êtes maintenant dans l’éditeur de programme pour le langage *littéral structuré*.

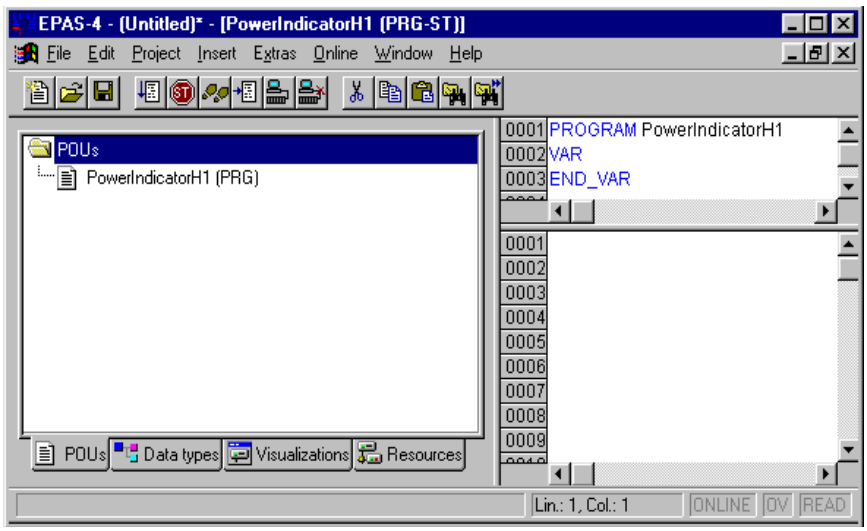



*Illustration 5-24: Editeur de programmes en littéral structuré*

- Entrez le programme et lancez la compilation en appuyant sur F11 (ou en passant par le menu PROJECT | COMPILE ALL).

PDM_UserMan_Richtlinien_fr.fm

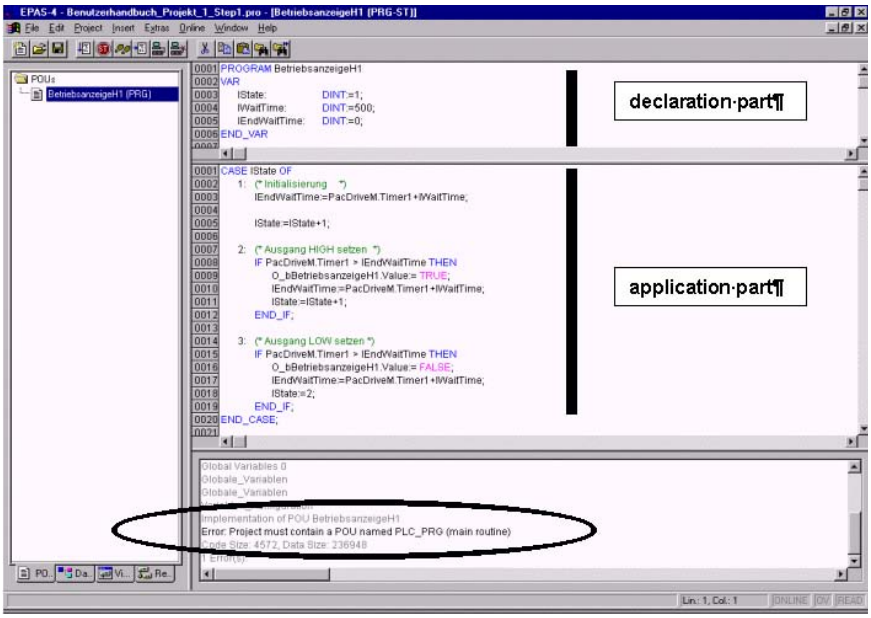



*Illustration 5-25: Entrée du programme utilisateur*

Un message d'erreur s'affiche.
Ce message d'erreur vous avertit qu'aucun programme n'a encore été spécifié dans la configuration de tâches.

- Sélectionnez « Resources » pour passer à la configuration de tâches.

*Illustration 5-26: Sélection de ressources dans l'organiseur d'objets*

- Sélectionnez *Task Configuration* pour atteindre l'éditeur correspondant.

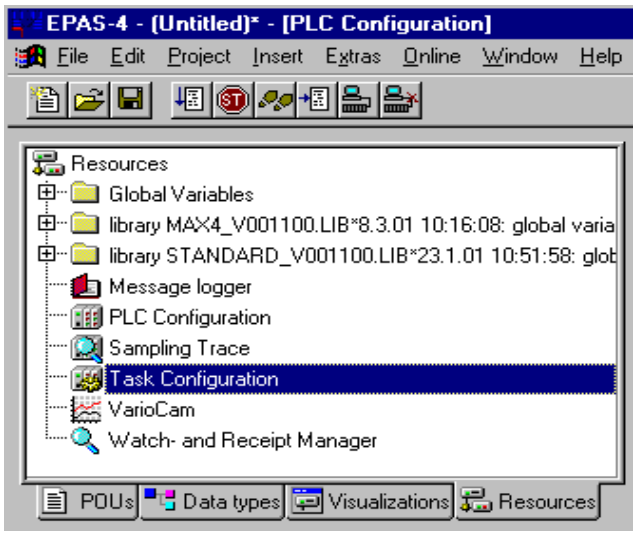



*Illustration 5-27: Sélection de la configuration des tâches*

Vous devez tout d’abord créer une tâche.

- Cliquez à droite et sélectionnez « Append Task ».

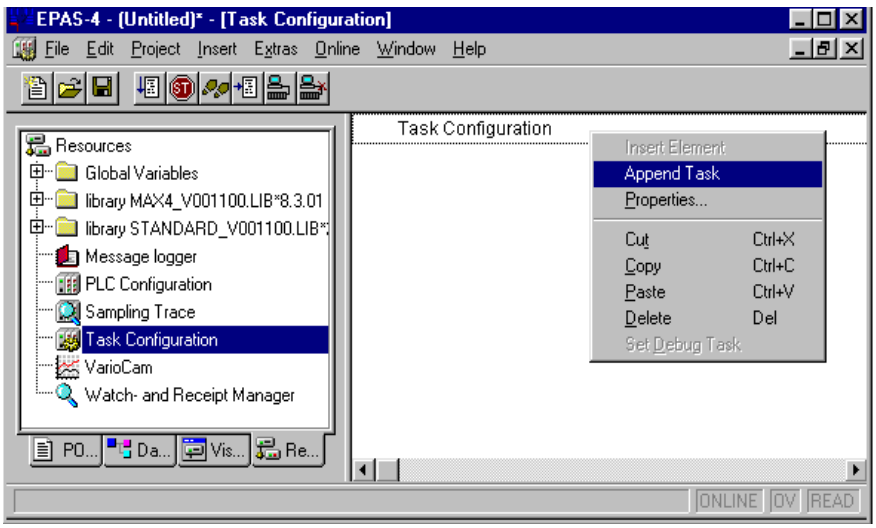



*Illustration 5-28: Ajouter une tâche*

- Identifiez la tâche en entrant un nom dans la boîte de dialogue. Conservez provisoirement la priorité de 31. Choisissez un intervalle de 100 ms.

PDM_UserMan_Richtlinien_fr.fm

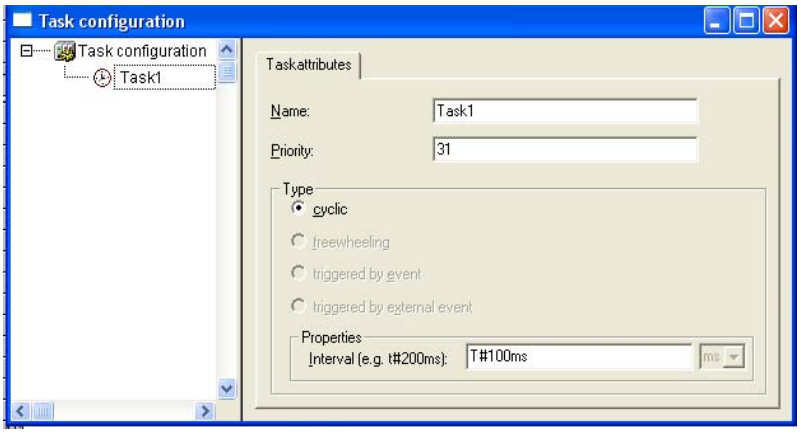



*Illustration 5-29: Définition des propriétés de la tâche*

- Confirmez vos entrées par « OK ».

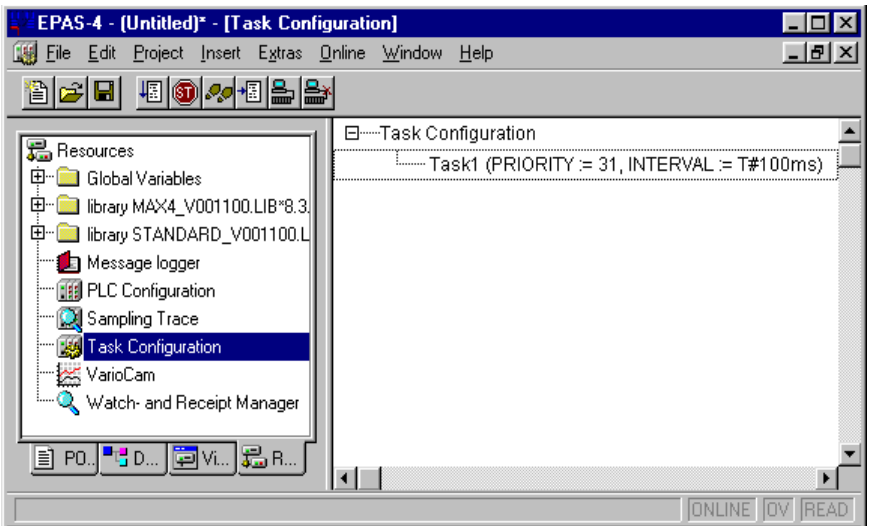



*Illustration 5-30: La configuration de tâches avec la nouvelle tâche*

Il vous reste à rattacher le programme créé à la tâche. Procédez comme suit :

- cliquez à droite sur Task1 (texte entre parenthèses)
- sélectionnez « Append Program Call »

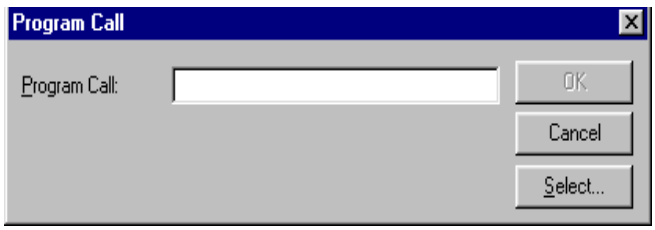



*Illustration 5-31: Boîte de dialogue « Program Call »*

PDM_UserMan_Richtlinien_fr.fm

- Affichez une sélection des programmes existants en cliquant sur « Select ».

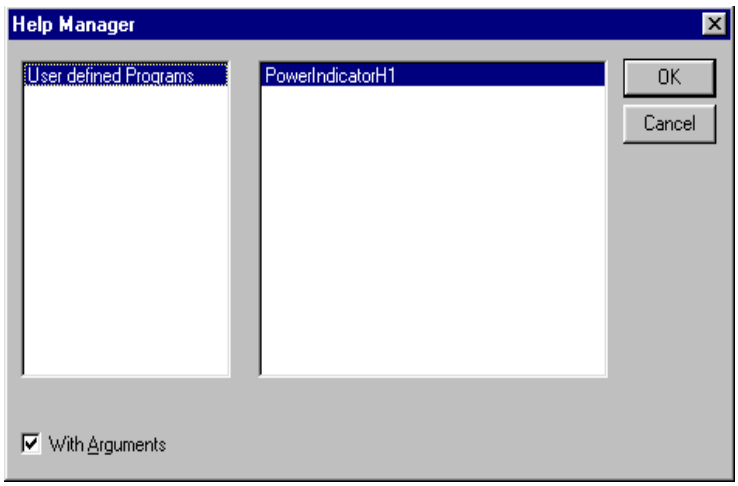



*Illustration 5-32: Assistant pour le dialogue « Program Call »*

- Sélectionnez le programme « PowerIndicatorH1 » et confirmez par OK.
- Confirmez votre sélection dans la boîte de dialogue « Program Call » en cliquant sur OK.

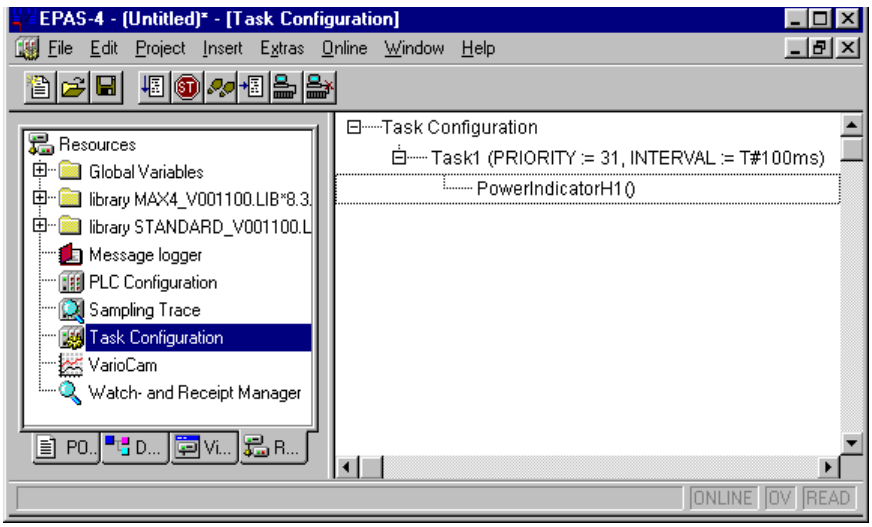



*Illustration 5-33: Configuration de tâches avec la nouvelle tâche et l’appel de programme*

- Revérifiez le projet avec « Compile all » (F11).

Plus aucun message d’erreur ne s’affiche ; le projet peut être transmis au PacDrive Controller.

## 5.7.5 Observer & Piloter

### Transmission d’un projet

- Dans EPAS-4, sélectionnez « Login » dans le sous-menu ONLINE pour transmettre le projet au PacDrive Controller. Vous pouvez vous connecter au PacDrive Controller en utilisant le symbole *Login*.
- Sélectionnez « Login » dans la boîte de dialogue ci-dessous.

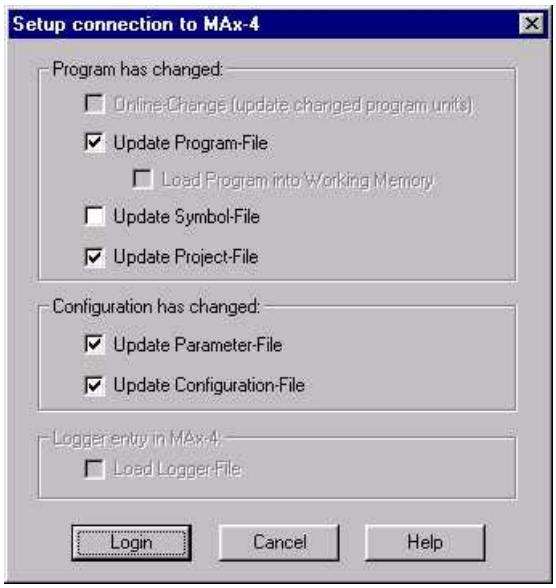



*Illustration 5-34: Boîte de dialogue pour la procédure d’identification et de mise en connexion*

Le programme est ensuite transmis au PacDrive Controller.
La configuration du PacDrive Controller ayant été modifiée, celui-ci doit être reinitialisé.

### Test d’un programme

Les fonctions suivantes sont disponibles dans le sous-menu ONLINE :

- Run
  Redémarrage du programme ou reprise à la prochaine instruction après une interruption.
- Stop
  Interruption du programme IEC.
- Reset
  Arrêt du programme IEC.

Il est également possible de programmer des points d’arrêt, par exemple, pour l’exécution d’une tâche en mode intermittent.

PDM_UserMan_Richtlinien_fr.fm

### Gestionnaire de surveillance et de recette

Le gestionnaire de surveillance et de recette vous permet de lister l’ensemble des variables et des paramètres d’objet importants pour les surveiller en ligne sans avoir à feuilleter ou ouvrir plusieurs fenêtres à la fois.

### Enregistrement de trace

Cette fonction permet d’enregistrer l’évolution des paramètres d’objet et des variables sur l’axe temps. Elle représente une sorte d’oscilloscope.

Pour enregistrer une trace, vous devez d’abord la configurer.

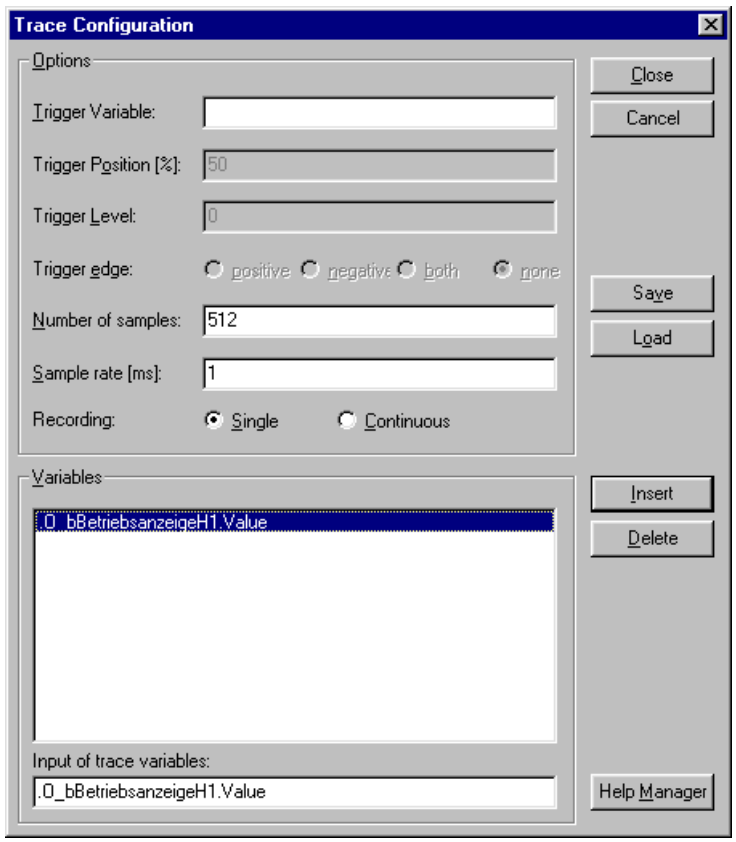



*Illustration 5-35: Boîte de dialogue « Trace Configuration » (Configuration de trace)*

A titre d’exemple, l’enregistrement de votre indicateur de marche ressemblera à ceci :

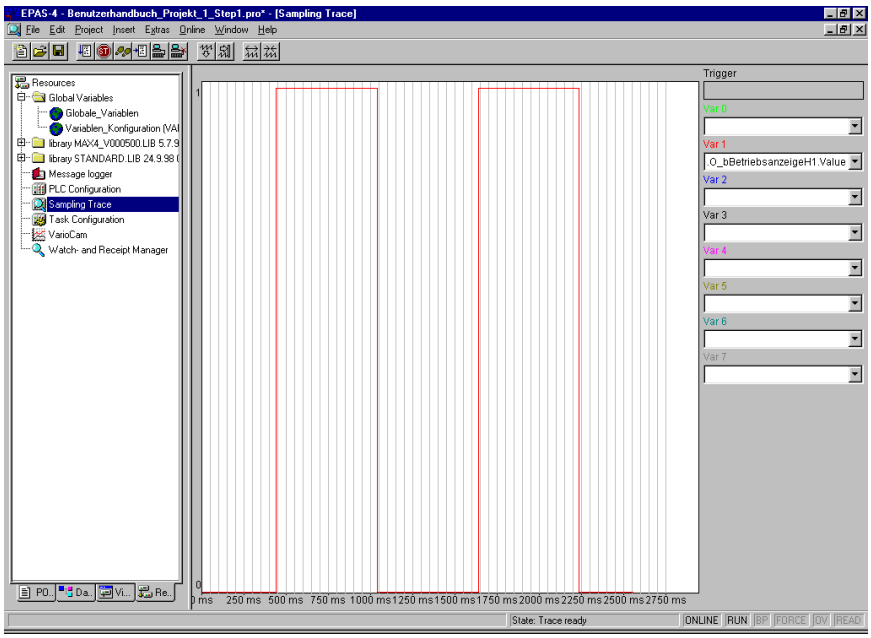



*Illustration 5-36: Enregistrement de la trace de l'indicateur de marche H1*

**REMARQUE**

Pour plus d'informations sur l'enregistrement de traces, veuillez consulter le Guide d'utilisation EPAS-4.

En regardant cet enregistrement de près vous remarquerez que la fréquence de clignotement de notre indicateur de marche n'est pas exactement de 1 Hz.

Comment ce léger écart s'explique-t-il ?

Le programme IEC que vous venez d'écrire est appelé toutes les 100 ms. Vous avez sélectionné cet intervalle dans la configuration des tâches.
La durée effective du « délai d'attente » de 500 ms peut donc varier entre 501 et 600 ms.

Mais cette variation est sans importance pour l'indicateur de marche puisque celui-ci ne commande qu'un voyant lumineux.

## 5.7.6 Documentation et sauvegarde d'un projet

La documentation de votre projet comprend les éléments suivants :

- cahier des charges
- fichier projet EPAS-4 et fichiers complémentaires éventuels
- versions logicielles et matérielles des équipements

PDM_UserMan_Richtlinien_fr.fm

- rapports d’essai
- procès-verbal des essais de réception
- sortie imprimante de la documentation à partir d’EPAS-4.

Veillez à choisir des noms de variable « parlants » et à commenter les blocs fonctionnels pour faciliter la maintenance de votre projet.

Toutes les modifications de programme doivent être signalées et repérables dans le nom du fichier de projet. Il suffit pour cela d’insérer la date ou la version dans le nom du projet.

Le sous-menu EPAS-4 PROJECT | DOCUMENTATION permet à l’utilisateur de mieux gérer la documentation.

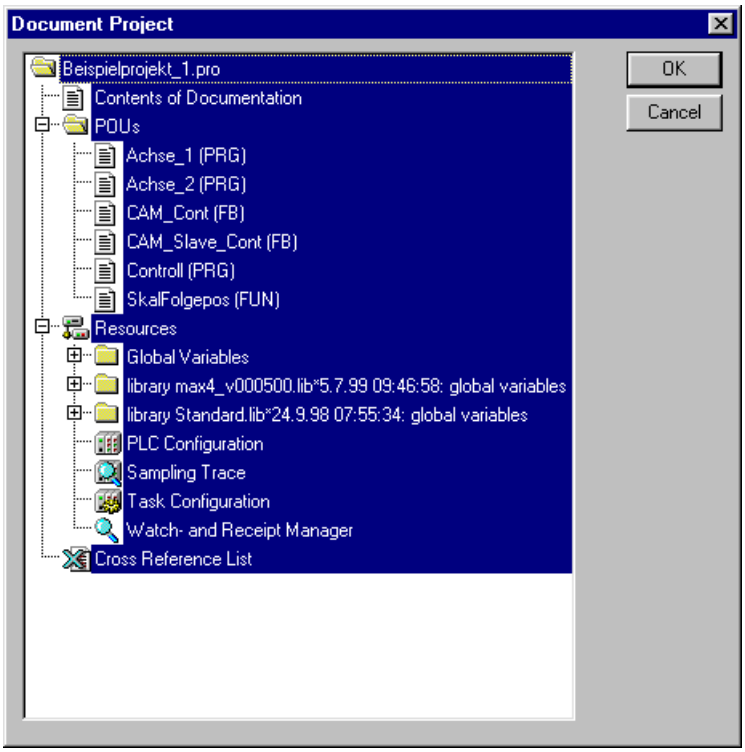



*Illustration 5-37: Boîte de dialogue EPAS-4 « Document Project »*

## 5.8 Elargissement de l’exemple par des fonctions de mouvement

### 5.8.1 Exposé du problème

Nous allons maintenant élargir notre projet « simple » par les fonctions de mouvement d’un moteur. Voici l’exposé du problème élargi :

Soit une installation dont l'indicateur de marche H1 doit clignoter dès qu'elle est mise en marche. Du matériel d'emballage doit être acheminé vers une cisaille dès que le bouton marche/arrêt de la bande transporteuse a été actionné. L'installation doit s'arrêter après une nouvelle utilisation du bouton marche/arrêt.

### Description d'un profil de mouvement

Le profil de mouvement de la bande transporteuse est composé d'une courbe de démarrage exécutée une fois après la mise en marche, d'une droite qui reste active tant que la bande est en marche et d'une courbe d'arrêt exécutée au moment de l'arrêt (bouton marche/arrêt sur arrêt).

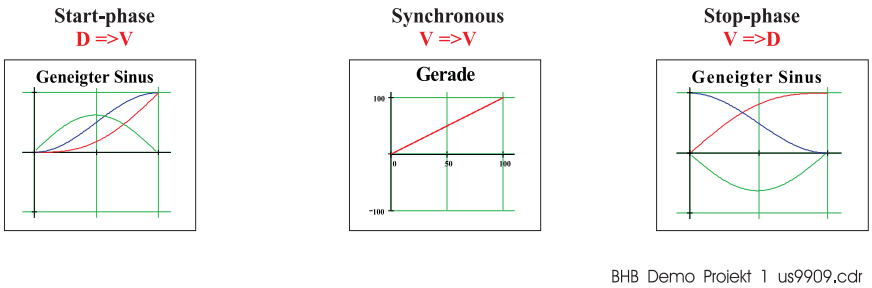



*Illustration 5-38: Profil de mouvement*

### Spécification des valeurs d'entrée et de sortie, respectivement des senseurs et actuateurs

| Equipements | Identification d'entrée/sortie | Commentaire |
|---|---|---|
| H1 | O_bEquipmentOnIndicatorH1 | Un voyant lumineux connecté à la sortie clignote avec une fréquence de 1 Hz dès que l'installation est mise en marche. |
| S1 | I_bStartS1 | Marche -> mise en marche / état de marche de l'installation<br>Arrêt -> mise à l'arrêt / état d'arrêt de l'installation |

*Tableau 5-9: Définition E/S*

PDM_UserMan_Richtlinien_fr.fm

**Représentation sur l’axe temps**

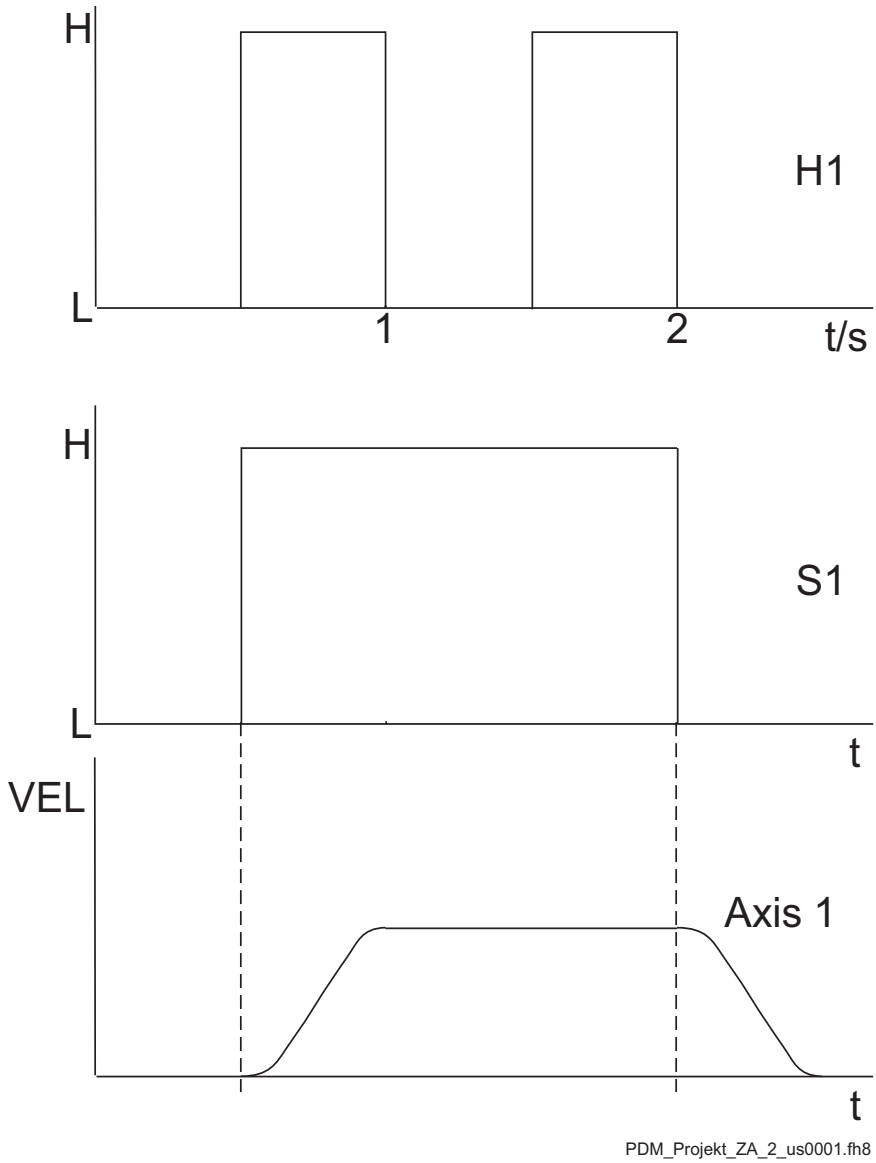



*Illustration 5-39: Evolution sur l’axe temps*

## 5.8.2 Matériel

La solution du problème nécessite l’association d’un PacDrive MC-4 au PacDrive Controller MAx-4.

## 5.8.3 Configuration

### Configuration d'un projet

Pour réaliser ce projet, il faut en plus un MotorController MC-4 avec servomoteur pour entraîner la bande transporteuse.

Le MotorController MC-4 est entré dans la configuration de commande :

- Cliquer à droite sur « Real time bus »

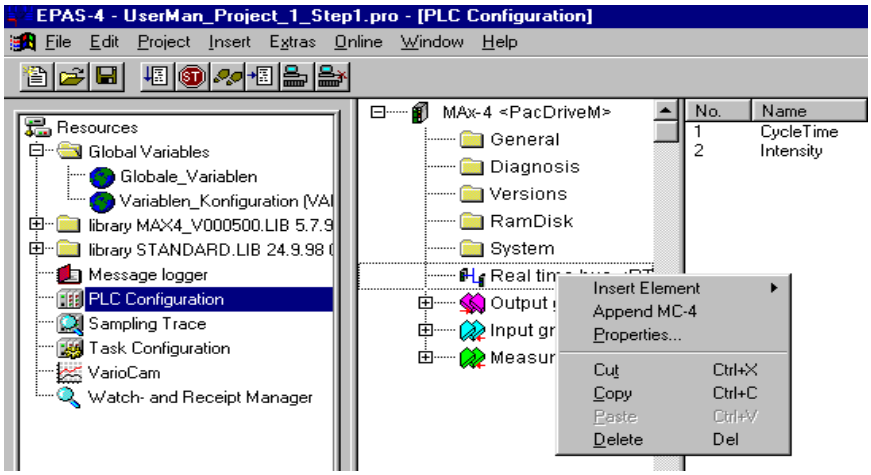



*Illustration 5-40: Entrée du MotorController MC-4 dans la configuration de commande*

- Sélectionner « Append MC-4 ».
- Spécifier le nom symbolique « conveyer_belt » pour MC-4.

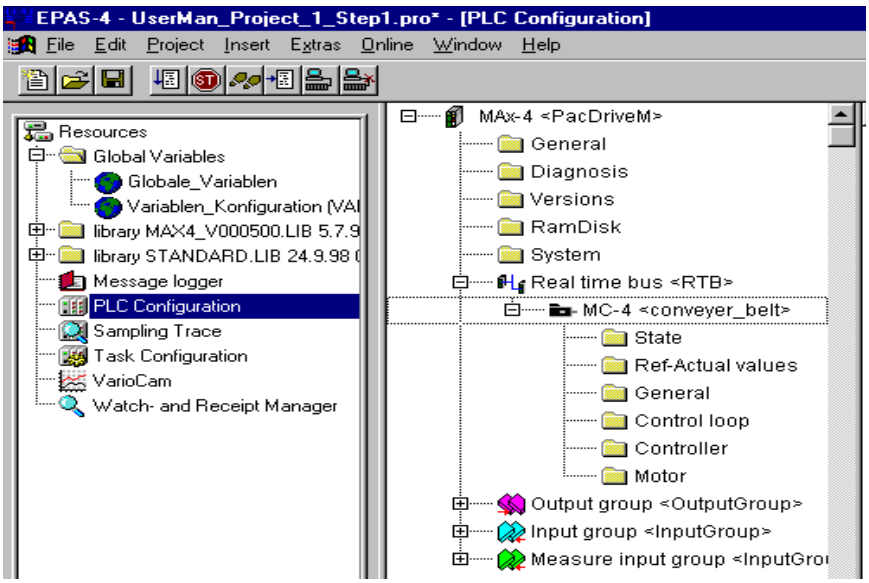



*Illustration 5-41: Attribution d'un nom symbolique pour le MotorController MC-4*

PDM_UserMan_Richtlinien_fr.fm

**REMARQUE**

Etant donné que nous allons d’abord « simuler » le MotorController MC-4 avec son servomoteur, il n’est pas encore nécessaire de procéder à d’autres réglages dans la configuration de commande.

Spécifions maintenant le nom symbolique de l’entrée S1. Nous avions choisi ce nom au moment de l’exposé du problème.

- Entrez le nom dans la configuration de commande.

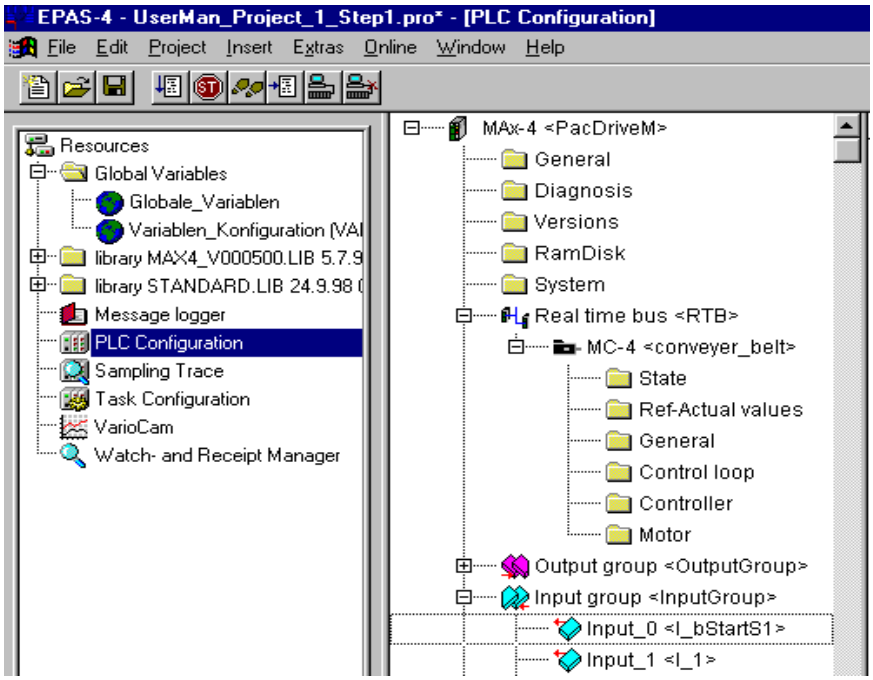



*Illustration 5-42: Attribution d’un nom symbolique à Input_0*

Le profil de mouvement étant composé de courbes, il faut un codeur maître.

**REMARQUE**

Chez ELAU, le principe de traitement et d’exécution des courbes s’appelle VarioCam™. Un axe (axe asservi) suit une position maîtresse (axe maître). La courbe spécifie la position de l’axe asservi par rapport à celle de l’axe maître.
Un codeur logique permet ensuite de transmettre aux fonctions de courbes les positions maîtresses spécifiées dans le programme IEC.

Le codeur logique est entré comme suit :

- Cliquer à droite sur MAx-4

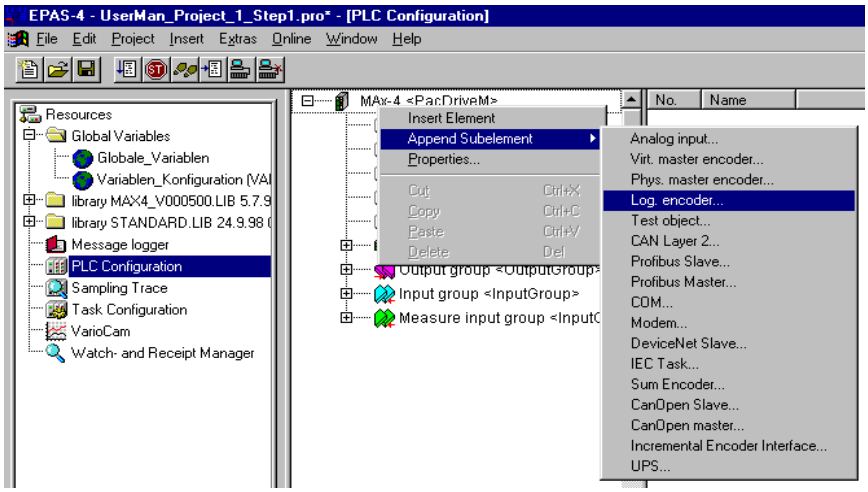



*Illustration 5-43: Entrée d'un codeur logique dans la configuration*

- Sélectionner le codeur logique « Log. encoder »
- Attribuez à ce codeur logique le nom symbolique « MasterEncoder ».

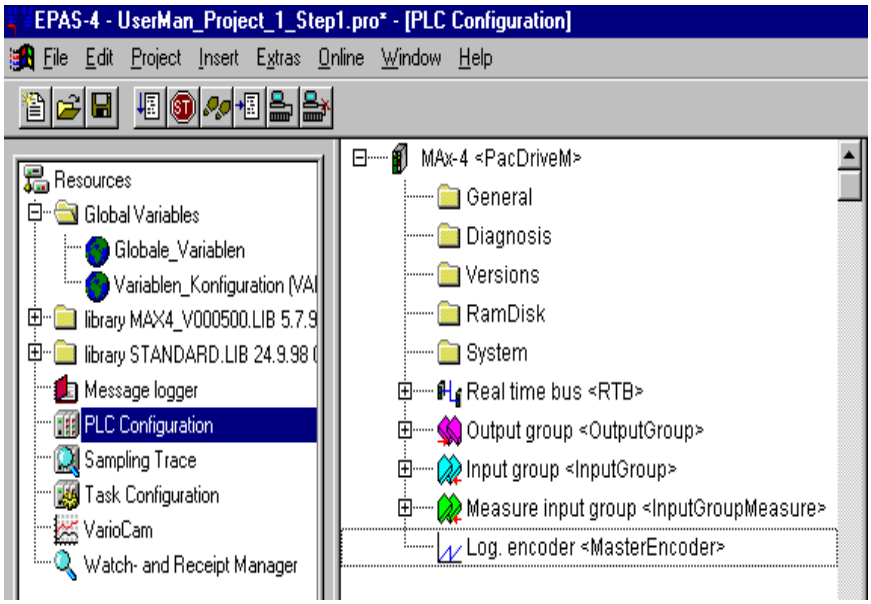



*Illustration 5-44: Attribution d'un nom symbolique au codeur logique*

## 5.8.4 Programme utilisateur

Créez maintenant un autre programme (cf. instructions ci-dessus) pour définir la fonction de mouvement de la bande transporteuse.

PDM_UserMan_Richtlinien_fr.fm

- Vous devez charger les profils des courbes pour les adapter au programme à réaliser :

```
(* load profiles *)
StartKurveId:=ProfilLoad('modisin');
ProfilSetLambda(StartCamId, 1.0);
(* -> dwell -> velocity *)

CyclicCamId:=ProfilLoad('straight');
(* -> velocity -> velocity *)

StopCamId:=ProfilLoad('modisin');
ProfilSetLambda(StopCamId, 0.0);
(* -> velocity -> dwell *)
```



**REMARQUE**

Vous trouverez une description détaillée des bibliothèques ELAU dans l’aide en ligne EPAS-4.

- Il s’agit maintenant de mettre chaque courbe à l’échelle :

```
(* scale axes ---------- *)
lrYFactor1:=ScalFollPos(1.0, g_lrPartLength,
g_K_MS);(* Start cam *)
lrYFactor2:=ScalFollPos(1.0, g_lrPartLength,
g_K_ST);(* Cyclic cam *)
lrYFactor3:=ScalFollPos(1.0, g_lrPartLength,
g_K_MS);(* Stop cam *)
```



**REMARQUE**

La fonction « ScalFollPos » calcule *lrYFactor*. Ce calcul consiste simplement dans l’opération suivante :
ScalFollPos:=(m * sMasterPos) / K;

- La programmation du mouvement proprement dit peut commencer :

```
(* start cam ---------------------------------- *)
4:(* prepare start cam *)
  bCamReset:=TRUE;
  g_lState_1:=g_lState_1+1;
5:(* start start cam *)
  CAM(lAxisId:= _ConveyerBelt, lEncId:=
_MasterEncoder,
  lProfilId:= StartCamId, lrXOffset:= 0.0,
  lrYOffset:= 0.0, lrXFactor:= g_lrPartLength,
  lrYFactor:= lrYFactor1,lrXLimMin:= 0.0,
  lrXLimMax:= g_lrPartLength,bXLimMinOn:= TRUE,
```

PDM_UserMan_Richtlinien_fr.fm

```
   bXLimMaxOn:= TRUE, iXSetposMode:= ABSOLUTE,
   lrXSetposPos:= 0.0, iYSetposMode:= ABSOLUTE,
   lrYSetposPos:= 0.0, bReset:= bCamReset);
   bCamReset:=FALSE;
   IF (CAM.lResult = 0) THEN
      Schlz_ZK:=0;
      bCamReset:=TRUE;
      g_lState_1:=g_lState_1+1;
   END_IF;
(* cyclic cam ---------------------------- *)
(* 1st cycle must have different position manipula-
tion! *)
6:(* start cyclic cam *)

   :
```

- Il vous reste à programmer la coordination du mouvement.

**REMARQUE**

Le projet modèle complet est disponible sur le CD EPAS-4.

- Créez une nouvelle *tâche* dans la *configuration de tâches* (« Task Configuration ») et entrez le programme « bande transporteuse ». Attention : la priorité de ce programme doit être supérieure à celle de l'indicateur de marche. Choisissez la priorité de 30 et un intervalle de 10 ms, de sorte que les courbes se suivent assez rapidement pour éviter les à-coups.

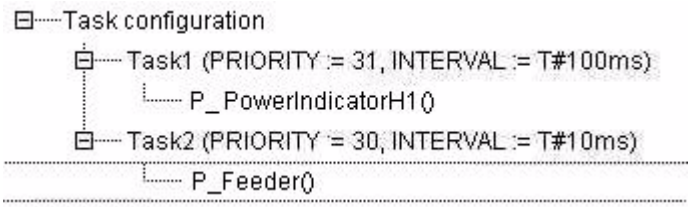



*Illustration 5-45: La configuration de tâches avec les deux tâches à priorité différente*

## 5.8.5 Observer & Piloter

### Transmission d'un projet

Identifiez-vous (cf. instructions ci-dessus) auprès du PacDrive Controller.

### Test d'un programme / Mise au point et test

Pour tester votre projet, plusieurs possibilités s'ajoutent à celles présentées dans le cadre de l'essai de la première version.

PDM_UserMan_Richtlinien_fr.fm

Mode intermittent

Il est possible d’exécuter le programme en mode intermittent. Pour ce faire, procédez comme suit :

- Spécifiez d’abord la tâche à déboguer. Pour ce faire, passez à la *configuration de tâches* (« Task Configuration ») et cliquez à droite sur la *tâche* que vous souhaitez déboguer.

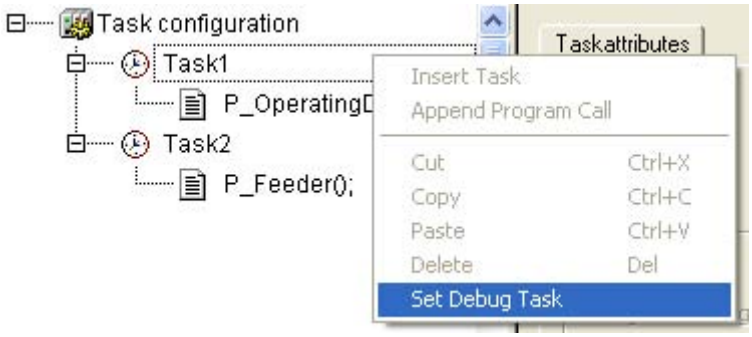



*Illustration 5-46: Spécification de la tâche à déboguer dans la configuration de tâches*

- Séle^1ctionnez « Set Debug Task ». Le repère [DEBUG] apparaît derrière le nom.

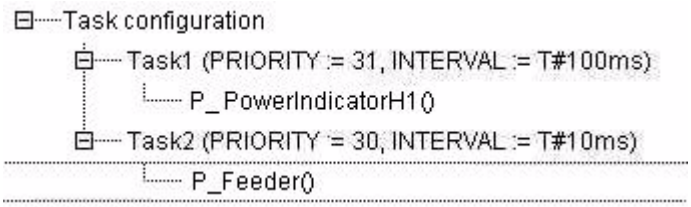



*Illustration 5-47: Affichage de la tâche à déboguer dans la configuration de tâches*

- Il faut maintenant placer un point d’arrêt. Le point d’arrêt marque un interruption dans l’exécution du programme. Pour insérer un point d’arrêt, allez dans une UOP et cliquez à gauche sur le numéro de ligne où le programme doit être interrompu. La boîte de dialogue de sécurité suivante est affichée :

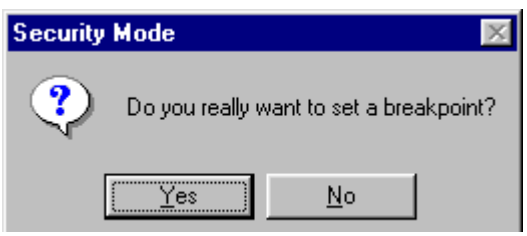



*Illustration 5-48: Boîte de dialogue de sécurité pour l’insertion d’un point d’arrêt*

- Confirmez en cliquant sur « Yes » pour placer le point d’arrêt. La ligne d’insertion du point d’arrêt est marquée d’une couleur différente.

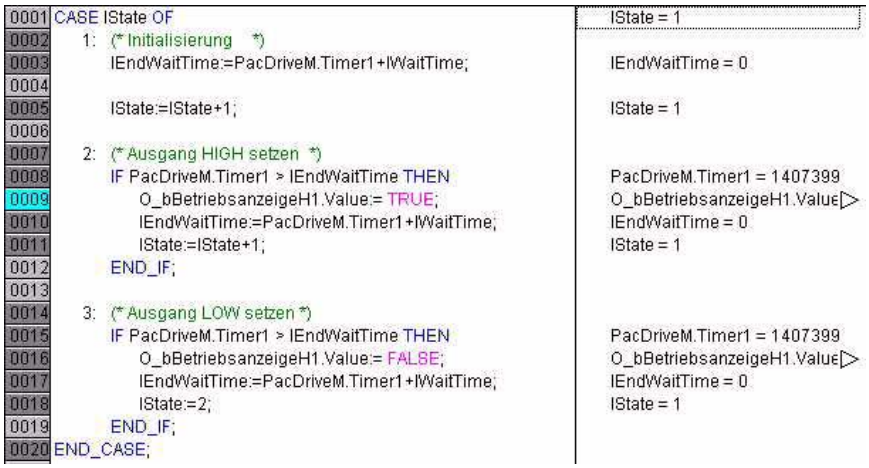



*Illustration 5-49: Editeur de programme ST en mode en ligne avec point d'arrêt*

- Lancez maintenant le programme. La couleur de marquage change dès que le point d'arrêt est atteint.

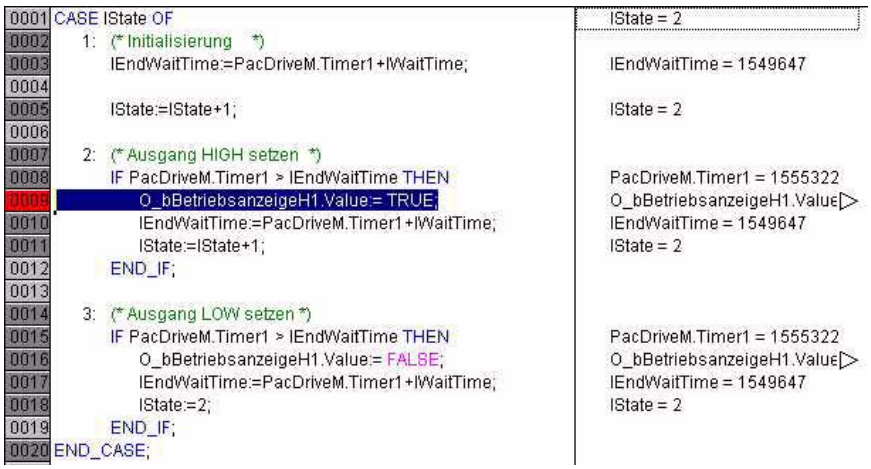



*Illustration 5-50: Le point d'arrêt est atteint*

- Vous pouvez maintenant relancer l'exécution du programme (ou de l'UOP) en mode *intermittent*. Lancez l'exécution individuelle d'un pas de programme avec le symbole ou avec les touches de fonction ou en passant par le sous-menu ONLINE.

PDM_UserMan_Richtlinien_fr.fm

*Illustration 5-51: Symbole du mode d’exécution intermittent et commandes correspondantes*

### Call stack… (Hiérarchie des appels)

Cette fonction permet d’afficher la pile des appels en cas de message de diagnostic transmis depuis l’enregistreur des messages.

Exemple :

- Nous provoquons une erreur en élargissant le programme comme suit :

```
0001 CASE IState OF
0002     1:  (* Initialisierung  *)
0003         IEndWaitTime:=PacDriveM.Timer1+IWaitTime;
0004
0005         IState:=IState+1;
0006
0007     2:  (* Ausgang HIGH setzen *)
0008         IF PacDriveM.Timer1 > IEndWaitTime THEN
0009             O_bBetriebsanzeigeH1.Value:= TRUE;
0010             IEndWaitTime:=PacDriveM.Timer1+IWaitTime;
0011             IState:=IState+1;
0012         END_IF;
0013
0014     3:  (* Ausgang LOW setzen *)
0015         IF PacDriveM.Timer1 > IEndWaitTime THEN
0016             O_bBetriebsanzeigeH1.Value:= FALSE;
0017             IEndWaitTime:=PacDriveM.Timer1+IWaitTime;
0018             IState:=2;
0019
0020             (* Fehler Division durch 0 auslösen! *)
0021             IEndWaitTime:=IEndWaitTime / 0;
0022
0023         END_IF;
0024 END_CASE;
0025
```

*Illustration 5-52: Programme ST avec erreur*

- Lancez le programme et attendez jusqu’à ce que le message de diagnostic soit transmis (LED d’erreur de MAx-4 commence à clignoter).
- Ouvrez l’enregistreur de messages et cliquez à droite sur le message « division error » (erreur de division).

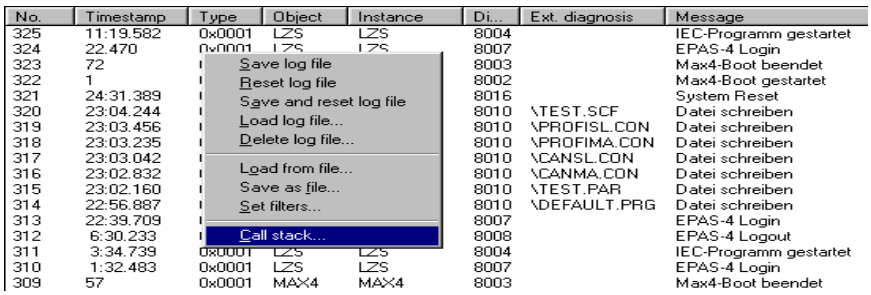

No. Timestamp Type Object Instance Di... Ext. diagnosis Message
325 11:19.582 0x0001 LZS LZS 8004 IEC-Programm gestartet
324 22.470 0x0001 LZS LZS 8007 EPAS-4 Login
323 72 8003 Max4-Boot beendet
322 1 8002 Max4-Boot gestartet
321 24:31.389 8016 System Reset
320 23:04.244 8010 \TEST.SCF Datei schreiben
319 23:03.456 8010 \PROFISL.CON Datei schreiben
318 23:03.235 8010 \PROFIMA.CON Datei schreiben
317 23:03.042 8010 \CANSL.CON Datei schreiben
316 23:02.832 8010 \CANMA.CON Datei schreiben
315 23:02.160 8010 \TEST.PAR Datei schreiben
314 22:56.887 8010 \DEFAULT.PRG Datei schreiben
313 22:39.709 8007 EPAS-4 Login
312 6:30.233 8008 EPAS-4 Logout
311 3:34.739 0x0001 LZS LZS 8004 IEC-Programm gestartet
310 1:32.483 0x0001 LZS LZS 8007 EPAS-4 Login
309 57 0x0001 MAX4 MAX4 8003 Max4-Boot beendet
Save log file
Reset log file
Save and reset log file
Load log file...
Delete log file...
Load from file...
Save as file...
Set filters...
Call stack...

*Illustration 5-53: Sélection de la hiérarchie des appels dans l'enregistreur de messages*

- Sélectionnez Call stack...

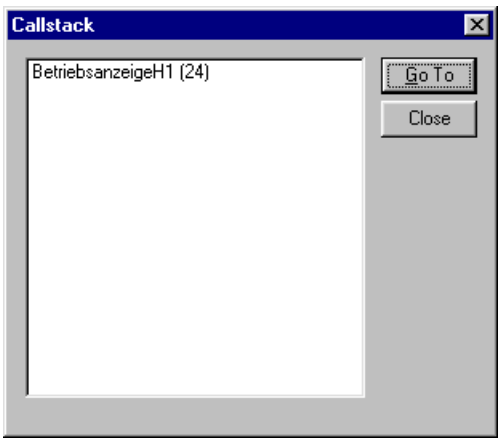



*Illustration 5-54: Boîte de dialogue « Callstack »*

Le dernier bloc fonctionnel est toujours celui où se trouve le pointeur d'exécution.

Un bloc fonctionnel est chargé dans une fenêtre après sélection de ce bloc et clic sur le bouton « Go To ». La ligne ou le réseau en cours d'exécution sont affichés.

## 5.8.6 Documentation et sauvegarde d'un projet

**REMARQUE**

N'oubliez pas de documenter et sauvegarder votre nouvelle version du projet.
Voir également « Documentation et sauvegarde d'un projet » à la page 130.

PDM_UserMan_Richtlinien_fr.fm

# 6 Stratégies de recherche d'erreurs

Le but de ce chapitre est de vous faciliter une détection d'erreur judicieuse et bien ciblée sur l'installation.

## 6.1 Marche à suivre

| | | |
|---|---|---|
| **Les DELs sur les appareils** | Les DELs permettent de tirer des premières conclusions concernant la cause de l'erreur. | voir section « DELs » |
| **Configuration de commande** | La configuration de commande permet une limitation continue de l'erreur à l'aide des paramètres objets. | voir section « Configuration de commande » |
| **Enregistreur de messages** | L'enregistreur de messages permet une analyse d'erreur détaillée.<br>A l'aide de l'histoire d'erreurs, aussi des erreurs consécutives peuvent être détectées avec certitude.<br>L'enregistreur de messages représente l'outil le plus important dans l'analyse d'erreurs sur le système PacDrive™. | voir section « Enregistreur de messages » |

*Tableau 6-1: Marche à suivre pour la première mise en service*

## 6.2 DELs

Les DELs permettent de tirer des premières conclusions concernant la cause de l'erreur.

Pour une description des DELs, voir le chapitre « Configuration / programmation / diagnostic » du guide d'utilisation respectif spécifique à l'appareil.

Documentation supplémentaire :

- Guide d'utilisation PacDrive Controller MAx-4
- Guide d'utilisation MotorController MC-4
- ...

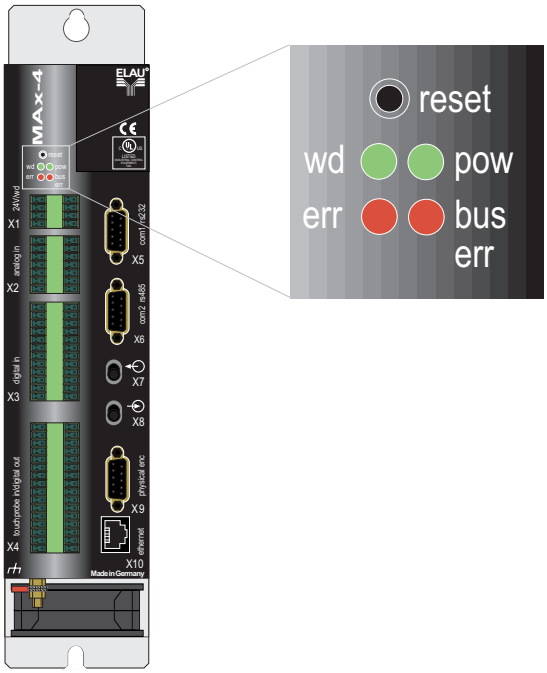



*Illustration 6-1: Exemple: DELs de diagnostic sur le PacDrive Controller MAx-4*

## 6.3 Configuration de commande

Les paramètres de diagnostic dans la configuration de commande permettent un diagnostic plus exact.

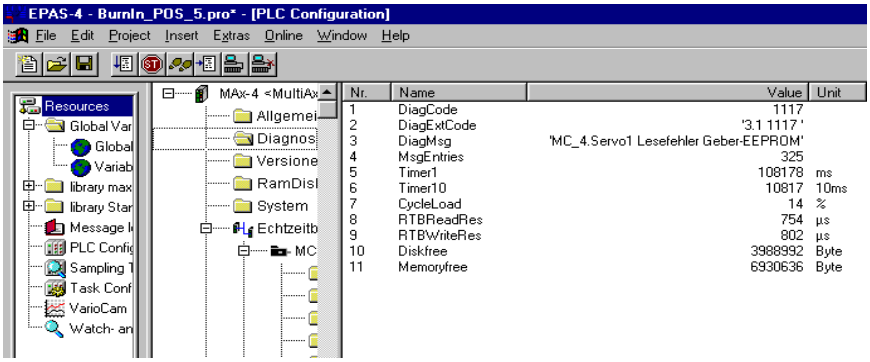



*Illustration 6-2: PLC Configuration (Configuration de commande)*

Le PacDrive Controller MAx-4 comprend les paramètres objet *DiagCode*, *DiagExtCode* et *DiagMsg*. Les paramètres contiennent l'information de diagnostic le plus important actuellement. Une superposition d'écriture n'a lieu que lors de l'apparence d'un message de diagnostic appartenant à une classe de diagnostic de priorité supérieure. Des messages de diagnostic appartenant à une classe de diagnostic de priorité inférieure ou de même niveau de priorité n'ont pas d'influence sur les paramètres. Il faut les confirmer pour effectuer une remise à l'état initial.

**Exemple :**

*DiagCode* 3117

*DiagExtCode* « 1.1 3117 »

*DiagMsg* « MC-4/axis1: motor temperature too high »

En cas d'erreurs spécifiques à l'axe (émetteur : MC_4) le placement des paramètres objet MotorController MC-4 *DiagCode* et *DiagMsg* est initié en plus. Les contenus des paramètres s'ensuivent en fonction des paramètres de diagnostic PacDrive Controller MAx-4. Leur actualisation suit les mêmes règles.

**Exemple :**

*DiagCode* 3117

*DiagMsg* « motor temperature too high »

**REMARQUE**

Pour plus d'informations, se reporter aux chapitres « Control Configuration » et « Diagnosis » dans l'aide en ligne EPAS-4.

# 6.4 Enregistreur de messages

## 6.4.1 Fonction

L'enregistreur de messages enregistre des messages de système, de diagnostic et d'application dans un tampon d'enregistrement des messages dans la zone RAM du PacDrive Controller MAx-4.

L'enregistrement s'effectue selon l'ordre chronologique dans un tampon en anneau (approx. 325 entrées). Elle comprend un horodateur (ms depuis lancement du système), le type de données, l'objet, l'instance, le code de diagnostic, le diagnostic ext. et le texte du message.

Le tampon d'enregistrement des messages ne pouvant être mémorisé au RAM que pendant 7 jours au maximum lors d'une coupure de courant, il peut être sauvegardé dans un fichier de consignation des messages sur le disque dans le PacDrive Controller MAx-4. Le PacDrive Controller MAx-4 est capable de sauvegarder plusieurs fichiers de consignation des messages sur son disque.

Les fichiers de consignation des messages peuvent être transmis du PacDrive Controller MAx-4 au PC à l'aide du logiciel de programmation EPAS-4. Ici aussi, plusieurs fichiers de consignation des messages peuvent être enregistrés.

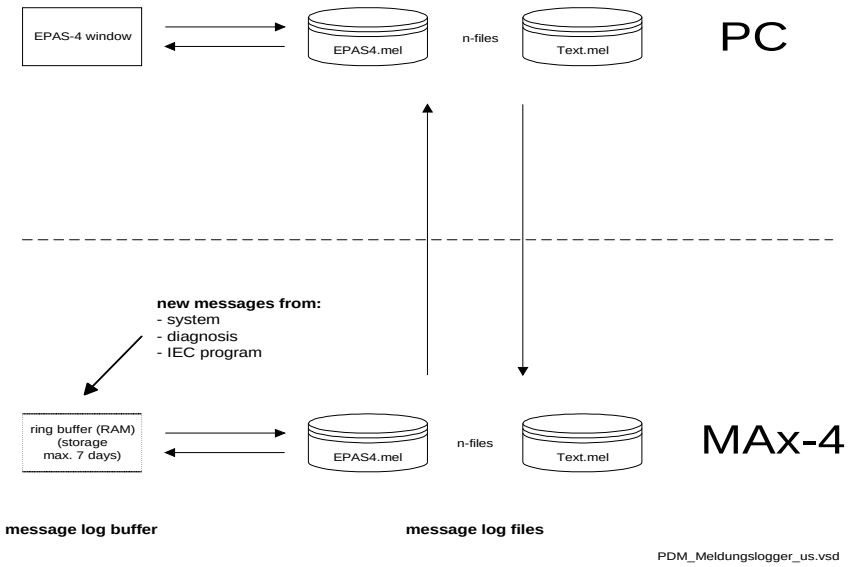



*Illustration 6-3: Ressources / enregistreur de messages*

## 6.4.2 Les fonctions individuels de l'enregistreur de messages

Save and load log file

Sauvegarder et charger l'enregistreur de messages actuel du PacDrive Controller MAx-4 dans le logiciel de programmation EPAS-4.

Save log file

Tampon d'enregistrement des messages (RAM)

-> Fichier de consignation des messages (fichier MAx-4)

Reset log entries

Ecraser le tampon d'enregistrement des messages (RAM)

Save and reset log file

Tampon d'enregistrement des messages (RAM)

-> Fichier de consignation des messages (fichier MAx-4)

-> Ecraser le tampon d'enregistrement des messages (RAM)

Load log file ...

Fichier de consignation des messages (fichier MAx-4)

-> Fichier PC

-> Affichage

Delete log file ...

Ecraser fichier MAx-4

Load from file ...

Fichier PC

-> Affichage

Save as file ...

Affichage

-> Fichier PC

| Fonctions | EPAS-4 / Enregistreur de messages (commande du menu Extras) | Programme IEC |
|---|---|---|
| Afficher l'enregistreur de messages actuel dans l'EPAS-4. | Save and load log file | |
| Save log file | Save log file (Sauvegarder le fichier de consignation) | MsgLogSave() |
| Reset log entries (Remettre les entrées du journal à l'état initial) | Reset log entries | MsgLogReset() |

| Fonctions | EPAS-4 / Enregistreur de messages (commande du menu Extras) | Programme IEC |
|---|---|---|
| Save and reset log file (Sauvegarder et remettre à l'état initial le fichier de consignation) | Save and reset log file | |
| Load log file ...(Charger le fichier de consignation) | Load log file ... | |
| Delete log file ... (Ecraser le fichier de consignation) | Delete log file ... | MsgLogDelete() |
| Load from file ... (Charger à partir du fichier ...) | Load from file ... | |
| Save as file ... (Sauvegarder comme fichier ...) | Save as file ... | |
| Entrée dans l'enregistreur de messages (RAM) | | MsgLogInsert() |

*Tableau 6-2: Fonctions de l'enregistreur de messages et modes d'accès*



**REMARQUE**

Il est également possible de filtrer l'enregistrement des messages (voir l'aide en ligne EPAS-4 / Control Configuration / MAx-4 / MsgFilter) ou bien de filtrer uniquement l'affichage des messages (voir chapitre suivant).

## 6.4.3 Filtrage des messages

Il y a deux manières de filtrer les messages système dans l'enregistreur de messages :

- Filtrage lors de l'enregistrement à partir du système via le paramètre objet MsgFilter dans la configuration de commande.
- Filtrage lors de l'affichage dans l'EPAS-4 via la commande du menu EXTRAS | SET FILTER...

L'enregistreur de messages distingue parmi 16 classes de message différentes. Il est possible d'activer et désactiver ces classes individuellement.

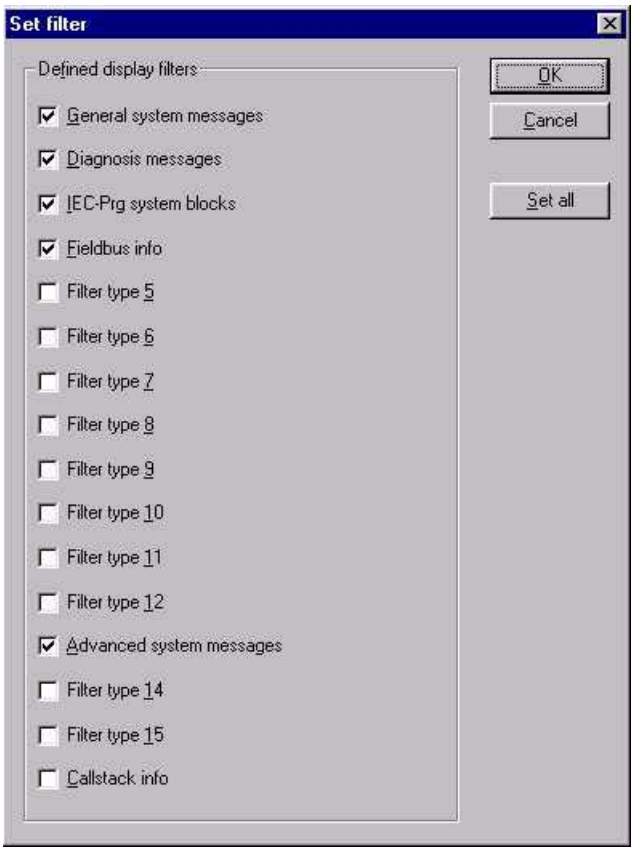



*Illustration 6-4: Ressources / enregistreur de messages / boîte de dialogue « Set filter » (filtrage des messages)*

**REMARQUE**

Les classes individuelles sont décrites dans l'aide en ligne EPAS-4 au chapitre « Control configuration ».

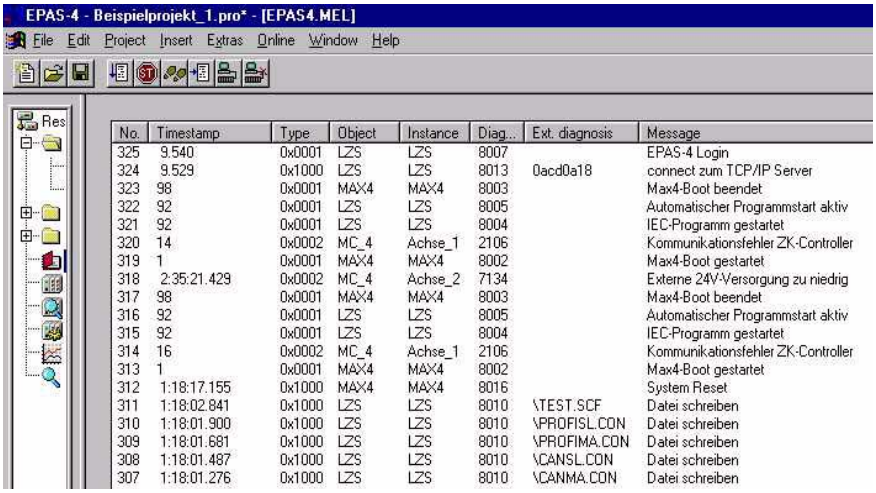
### 6.4.4 Etablir la liaison avec le PacDrive Controller MAx-4 en cas d'erreur

Lorsque vous établissez la liaison avec le PacDrive Controller MAx-4 en cas d'erreur, une boîte de sélection est affichée.

*Illustration 6-5: Ressources / enregistreur de messages / boîte de dialogue « Setup connection to MAx-4 » pour établir la liaison en cas d'erreur*

En cas d'erreur, l'option Load Logger-File est cochée.

- Lorsque vous sélectionnez Login, le fichier de consignation des messages est chargé à partir du PacDrive Controller MAx-4 dans EPAS-4 et y affiché.




| No. | Timestamp | Type | Object | Instance | Diag.. | Ext. diagnosis | Message |
|---|---|---|---|---|---|---|---|
| 325 | 9.540 | 0x0001 | LZS | LZS | 8007 | | EPAS-4 Login |
| 324 | 9.529 | 0x1000 | LZS | LZS | 8013 | 0acd0a18 | connect zum TCP/IP Server |
| 323 | 98 | 0x0001 | MAX4 | MAX4 | 8003 | | Max4-Boot beendet |
| 322 | 92 | 0x0001 | LZS | LZS | 8005 | | Automatischer Programmstart aktiv |
| 321 | 92 | 0x0001 | LZS | LZS | 8004 | | IEC-Programm gestartet |
| 320 | 14 | 0x0002 | MC_4 | Achse_1 | 2106 | | Kommunikationsfehler ZK-Controller |
| 319 | 1 | 0x0001 | MAX4 | MAX4 | 8002 | | Max4-Boot gestartet |
| 318 | 2:35:21.429 | 0x0002 | MC_4 | Achse_2 | 7134 | | Externe 24V-Versorgung zu niedrig |
| 317 | 98 | 0x0001 | MAX4 | MAX4 | 8003 | | Max4-Boot beendet |
| 316 | 92 | 0x0001 | LZS | LZS | 8005 | | Automatischer Programmstart aktiv |
| 315 | 92 | 0x0001 | LZS | LZS | 8004 | | IEC-Programm gestartet |
| 314 | 16 | 0x0002 | MC_4 | Achse_1 | 2106 | | Kommunikationsfehler ZK-Controller |
| 313 | 1 | 0x0001 | MAX4 | MAX4 | 8002 | | Max4-Boot gestartet |
| 312 | 1:18:17.155 | 0x1000 | MAX4 | MAX4 | 8016 | | System Reset |
| 311 | 1:18:02.841 | 0x1000 | LZS | LZS | 8010 | \TEST.SCF | Datei schreiben |
| 310 | 1:18:01.900 | 0x1000 | LZS | LZS | 8010 | \PROFISL.CON | Datei schreiben |
| 309 | 1:18:01.681 | 0x1000 | LZS | LZS | 8010 | \PROFIMA.CON | Datei schreiben |
| 308 | 1:18:01.487 | 0x1000 | LZS | LZS | 8010 | \CANSL.CON | Datei schreiben |
| 307 | 1:18:01.276 | 0x1000 | LZS | LZS | 8010 | \CANMA.CON | Datei schreiben |

*Illustration 6-6: Ressources / enregistreur de messages / fichier de consignation des messages*

PDM_UserMan_Fehlersuche_fr.fm

**Signification des entrées individuelles dans l'enregistreur de messages**

No.

Numéro consécutif des entrées enregistreur. Le numéro le plus haut correspond à l'entrée enregistreur la plus récente.

Timestamp

Temps écoulé depuis la mise en marche du PacDrive Controller MAx-4, format :

JJJ HH:MM:SS.sss

avec

- JJJ - jours
- HH - heures (0 ... 23)
- MM - minutes (0 ... 59)
- SS - secondes (0 ... 59)
- sss - millisecondes (0 ... 999)

Aux versions < 00.07.00, le temps est représenté en ms.

Type

Ici figure le type de filtrage du message.

Voir également l'aide en ligne EPAS-4 / Control configuration MAx-4.General.MsgFilter.

Object

Ici figure le type de l'objet responsable pour l'entrée dans l'enregistreur.

Voir également l'aide en ligne EPAS-4 / Control configuration.

Instance

Ici figure l'instance de l'objet responsable pour l'entrée dans l'enregistreur.

Voir également l'aide en ligne EPAS-4 / Control configuration.

Diag. Code

Ici figure le code de diagnostic.

Voir également l'aide en ligne EPAS-4 / Diagnosis.

Ext. diagnosis

Cette colonne offre des informations supplémentaires optionnelles sur le code de diagnostic. Dans la configuration de commande, dans MAx-4.diagnosis.DiagExtCode, ces informations sont aussi affichées en tant qu'information supplémentaire optionnelle.

Message

Ici figure le texte de message appartenant au code de diagnostic.

## 6.5 Autres moyens de diagnostic

**REMARQUE**

Pour plus d'informations, se reporter au chapitre « Diagnosis and Maintenance (Diagnostic et Maintenance) » dans les Modes d'emploi et au chapitre « Diagnosis » dans l'aide en ligne EPAS-4.

### 6.5.1 Trace

Cet enregistrement sert à enregistrer les relations temporelles et le comportement dynamique du système.

**REMARQUE**

Pour plus d'informations, se reporter au mode d'emploi EPAS-4.

### 6.5.2 Mise au point et test

Le système PacDrive™ offre à l'utilisateur des possibilités confortables pour la mise au point et le test de programmes.

Voir également « Test d’un programme / Mise au point et test » à la page 138.

### 6.5.3 La bibliothèque Check

En cas d'intégration de la bibliothèque « Check » dans le projet, lors de chaque accès sur un élément de tableau il y a un appel automatique du bloc fonctionnel Check qui vérifie les limites de plage de l'élément de tableau. De même, il y a une reconnaissance d'erreur de division (division par 0).

**REMARQUE**

L'intégration de la bibliothèque « Check » augmente la durée de traitement pour la raison indiquée ci-dessus. Il est donc recommandé d'éliminer la bibliothèque « Check » du projet après la terminaison de la phase de test.

### 6.5.4 Hiérarchie d'appels (CallStack)

Vous pouvez lancer la commande CallStack (« Online > Call stack... ») lors de l'arrêt du projet à un point d'arrêt. Une boîte de dialogue s'ouvre avec une liste des blocs fonctionnels se trouvant actuellement dans la pile d'appels.

Exemple pour une hiérarchie d'appels

Le premier bloc fonctionnel est toujours PC_PRG, marquant le début de l'exécution.

Le dernier bloc fonctionnel est toujours le bloc où l'exécution s'est arrêtée actuellement.

Après sélection d'un des blocs fonctionnels et actionnement du bouton « Go To », le bloc sélectionné est chargé dans une fenêtre affichant la ligne ou le réseau comprenant l'exécution.

# 7 Télémaintenance

## 7.1 Sommaire

La télémaintenance a pris une importance considérable au cours des dernières années dans l'industrie des équipements. Une telle tendance résulte non seulement des **T**echnologies d'**I**nformation et de **C**ommunication (TIC) actuelles appliquées à la télémaintenance (maintenance à distance), mais aussi de la possibilité de générer de nouveaux critères d'installation universelle sur l'ensemble des marchés.
La télémaintenance ne se limite pas uniquement à la correction à distance des erreurs. En effet, la mise en oeuvre des technologies de télémaintenance apporte un nouvel élan prometteur en matière de service après-vente (« After-Sales-Service ») et ce, pour toute la durée de vie d'une machine d'emballage. Cela englobe désormais la « télé-ingénierie » et le « télé-consulting ».

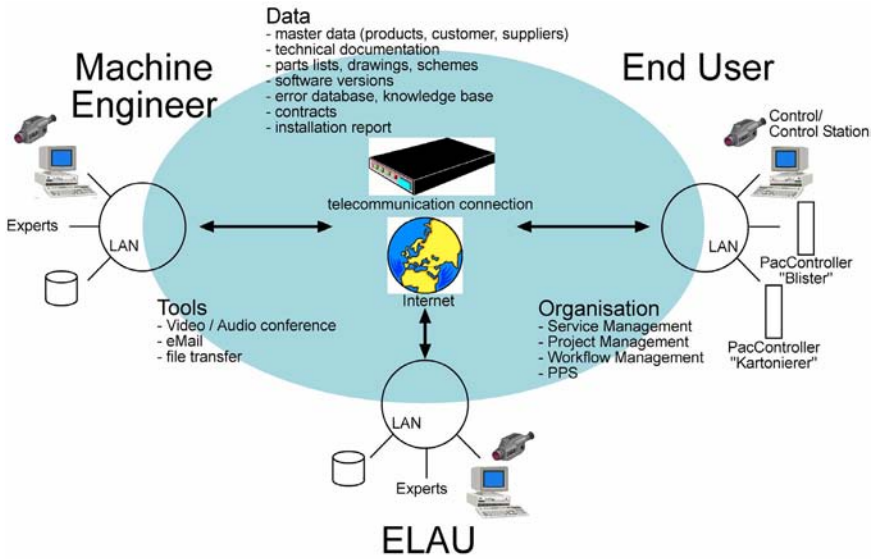



*Illustration 7-1: Concept de la télémaintenance*

### Définition

La télémaintenance est considérée comme support du service après-vente et support de l'application ; elle utilise des composantes et des prestations TIC qui permettent d'effectuer à distance des diagnostics et des corrections d'erreurs sur les machines. La télémaintenance est mise en oeuvre lors de l'installation et de la mise en marche de machines et d'équipements, lors de la correction des défauts et pour la transmission de nouvelles versions de logiciels. L'avenir de la

PDM_UserMan_Teleservice_fr.fm

télémaintenance réside, entre autres, dans le support de processus et le conseil client.

### Raccordement de l'automate PacDrive Controller MAx-4

Le PacDrive Controller MAx-A (= serveur) dispose d'une interface RS232 ainsi que d'une interface Ethernet. L'interface RS232 permet une communication série (raccordement d'un câble zéro modem ou d'un modem). L'interface Ethernet permet de connecter l'automate directement à un réseau local (LAN, Local Area Network) ou étendu (WAN, Wide Area Network) via un réseau TCP/IP. L'ordinateur (= client) est équipé également d'un port COM et d'une carte Ethernet avec la prise correspondante. Ainsi, de part et d'autre, les interfaces physiques sont implantées pour la liaison avec le monde extérieur.

Il existe plusieurs possibilités de connexion au système PacDrive™ via un client :

- Réseau local Ethernet,
- Connexion à distance (via un modem) et
- Connexion Internet.

Réseau local Ethernet

Pour effectuer un télédiagnostic sur l'automate, la première possibilité est la liaison Ethernet. Comme le montre la figure ci-dessous, le serveur et le client doivent obligatoirement être reliés par un réseau TCP/IP. Le réseau peut être installé par exemple dans l'entreprise et permet de mettre plusieurs ordinateurs en réseau.

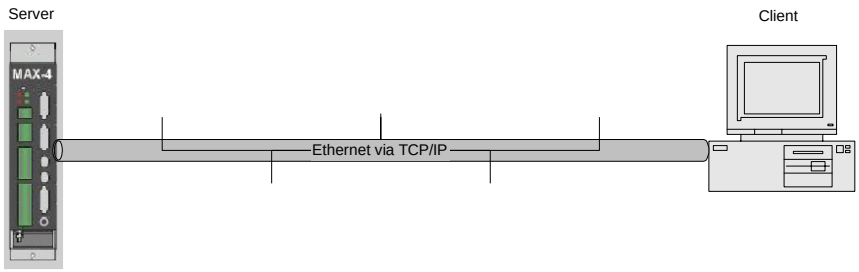



*Illustration 7-2: Réseau local Ethernet*

Cette méthode utilise le protocole TCP/IP. Les deux ordinateurs doivent avoir chacun une adresse IP. Cette adresse sera établie d'une façon univoque dans le réseau.

La connexion via un réseau local Ethernet est supportée par défaut par EPAS-4. C'est pourquoi ce point ne sera pas approfondi.

Connexion à distance directe

L'atelier logiciel EPAS-4 permet de créer une liaison directe par modem vers un automate.

Avantage :
Accès à toutes les fonctions du système.

Inconvénient :
Seuls deux ordinateurs peuvent être reliés directement. EPAS-4 doit obligatoirement être installé sur l'ordinateur client. La plate-forme n'est pas indépendante.

Connexion Internet

La télémaintenance doit pouvoir garantir un accès mondial au PacDrive Controller MAx-4. La toile (World Wide Web) est le moyen idéal. Quel que soit l'endroit, avec un ordinateur, un modem et un raccordement au réseau téléphonique, l'accès à l'Internet est immédiat.

Avantage :
L'utilisateur accède au service de diagnostic à distance indépendamment d'un lieu ou d'une plate-forme. Pas besoin d'EPAS-4.

Inconvénient :
L'accès au système est beaucoup plus restreint.

## 7.2 Connexion à distance au PacDrive Controller MAx-4

Veuillez impérativement tenir compte de l'indication suivante :

**REMARQUE**

Le modem étant raccordé sur le port série COM1 de l'automate PacDrive MAx-4, il n'est pas possible d'avoir en même temps une liaison série pour l'atelier logiciel EPAS-4. Dans le cas d'une télémaintenance via un modem, il faut par conséquent disposer en permanence d'une liaison TCP/IP opérationnelle pour EPAS-4.

### 7.2.1 Principe de fonctionnement

La télémaintenance via une connexion à distance directe, au moyen d'un modem et d'EPAS-4, permet à l'utilisateur d'accéder en principe à toutes les fonctions habituellement disponibles en présence d'une liaison physique directe.

**AVERTISSEMENT !**

Surveillance réduite auprès de la machine !

Risques de dommages corporels et matériels !

- La télémaintenance sera effectuée uniquement s'il y a un opérateur disponible auprès de la machine et si cet opérateur est qualifié.
- Il doit y avoir un contact permanent (par téléphone ou vidéo-conférence) avec l'opérateur.

Le système doit être configuré de telle sorte qu'il y ait une prise pour un modem (côté client et côté serveur).

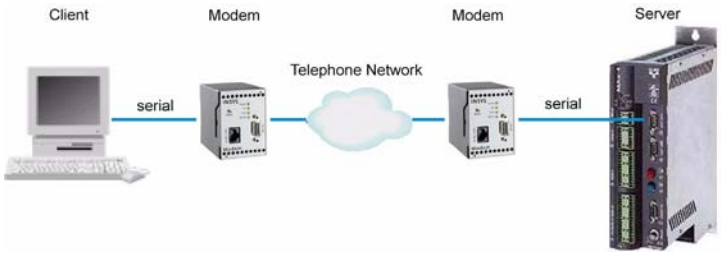



*Illustration 7-3: Connexion à distance directe*

Il est possible d'utiliser un modem analogique ou un modem ISDN. Dans le cas d'un modem ISDN, il faut veiller à ce que le driver normal ISDN ne soit pas activé dans l'ordinateur client : un émulateur modem analogique devra être installé. Le raccordement au serveur exige un modem analogique car l'ISDN n'est pas encore supporté.

Dans cette configuration, le protocole TCP/IP doit être obligatoirement installé des deux côtés. Le client et l'automate doivent avoir une adresse IP. L'utilisateur peut définir lui-même les adresses. Du fait que seuls ces deux ordinateurs sont reliés entre eux, il n'est pas nécessaire de vérifier si les adresses IP ont déjà été attribuées à d'autres ordinateurs.

Le modem est branché sur le port série COM1 du PacDrive Controller MAx-4. Pour démarrer le serveur PPP qui assure le contrôle de la connexion dans le PacDrive Controller Max-4, un projet doit être chargé qui comportera un objet modem ; le paramètre *Com1User* du PacDrive Controller MAx-4 doit avoir comme valeur « Modem / 1 » ou « Auto / 2 ».

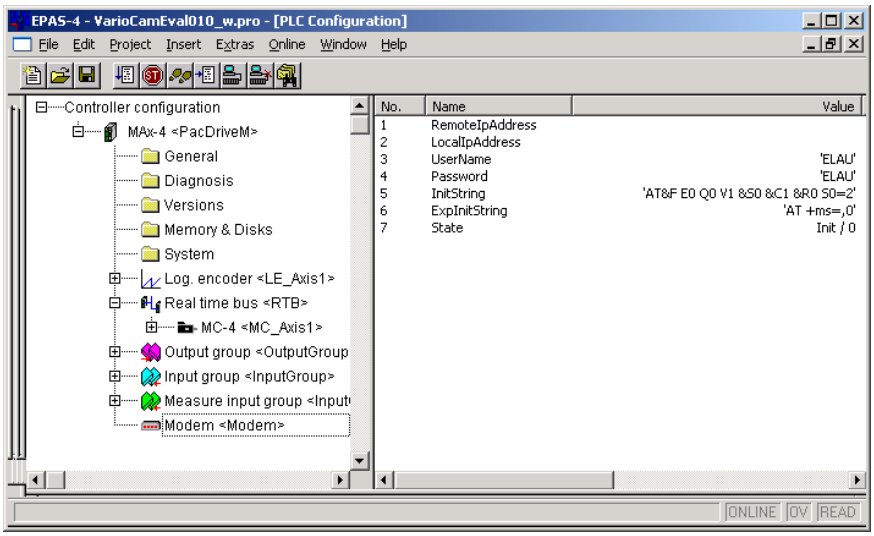



*Illustration 7-4: Configuration du modem*

Le modem est automatiquement reconnu lors du démarrage de l'automate s'il est allumé pendant ce démarrage et raccordé au port COM. Si le système ne détecte pas de modem allumé, le port Com est libéré pour la liaison série de l'atelier logiciel EPAS-4.
Voir également « Paramètres du projet pour le PacDrive Controller MAx-4 » à la page 162.

Le PacDrive Controller Max-4 contrôle le port série jusqu'à ce que le modem réponde à un appel entrant et prenne alors en charge l'établissement de la connexion.

L'appel est initialisé par le réseau d'accès à distance de l'ordinateur distant. Au moment de la numérotation, l'utilisateur doit obligatoirement entrer le nom d'utilisateur et le mot de passe choisi pour l'objet modem dans la configuration de l'automate afin de pouvoir établir une connexion. Lorsque le mot de passe et le nom d'utilisateur correspondent avec ceux prédéfinis, la connexion est établie. Celle-ci peut ensuite être traitée sur l'ordinateur distant relié à l'EPAS-4 comme une liaison TCP/IP normale vers le PacDrive Controller MAx-4.

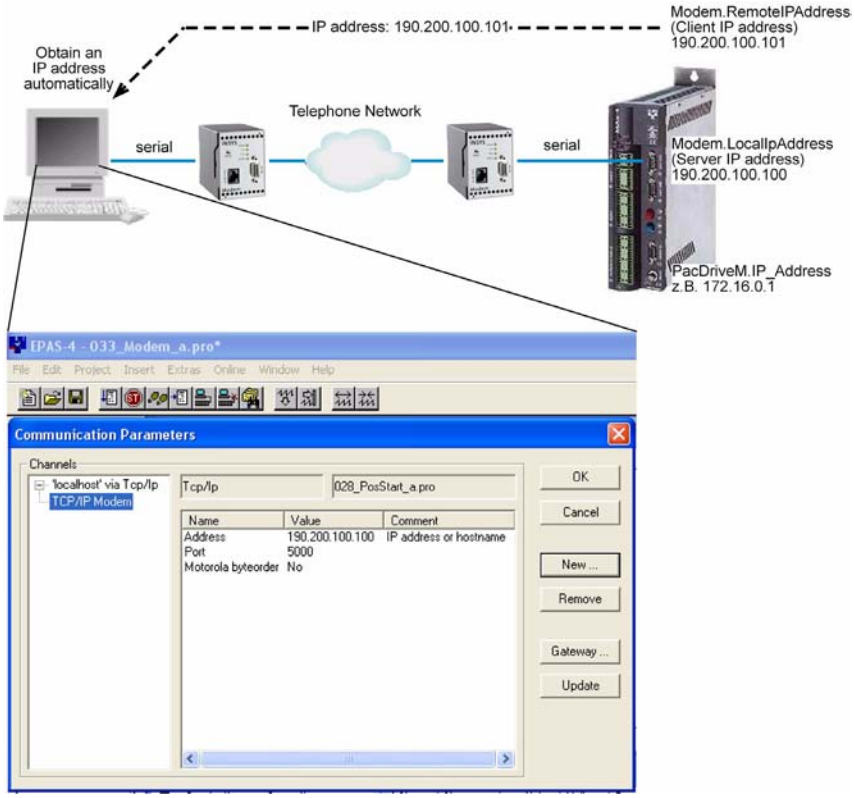



*Illustration 7-5: Schéma d'attribution des adresses IP*

Côté client, il faut un réseau d'accès à distance pour obtenir l'accès au serveur. Le réseau est configuré à l'aide d'un assistant. Les instructions détaillées ne font pas l'objet du présent manuel ; il convient de consulter le cas échéant l'aide en ligne de Windows. Ici, seuls les paramétrages les plus importants sont indiqués. Il faut entrer le numéro de téléphone du modem raccordé au serveur ainsi que l'adresse IP du serveur.

Si le client appelle le serveur via le modem, un nom d'utilisateur et un mot de passe sont demandés par sécurité. Ces paramètres peuvent être configurés sous EPAS-4.

## 7.2.2 Choix du modem

Le modem utilisé pour le raccordement à l'automate PacDrive Controller MAx-4 doit obligatoirement être compatible Hayes et supporter les commandes AT. Il faut toutefois vérifier que les paramètres de base suivants sont rentrés dans la configuration par défaut :

- Mode réponse automatique « Auto-Answer » (AT S0=2 : le modem décroche après la deuxième tonalité.)

- Traitement du signal DCD (AT &C1 : le signal DCD suit le niveau de la porteuse.)
- Activation du handshake matériel (AT &R0 : le modem réagit au signal CTS.)
- Désactivation de la fonction écho (AT E0)
- Activation comptes rendus (AT Q0)
- Messages en verbal (AT V1)

L'interface série est paramétrée pour la communication modem comme suit :

- Vitesse de transmission : 38400
- Bit de données : 8
- Bit d'arrêt : 1
- Parité : aucune

Lors du démarrage, le PacDrive Controller MAx-4 cherche à initialiser le modem selon ces paramètres. Il utilise le paramètre *InitString* du modem. Les commandes AT adéquates doivent alors être entrées pour régler les paramètres de base. La chaîne d'initialisation par défaut contient normalement les commandes nécessaires pour la plupart des modems. Il conviendra toutefois de vérifier dans la documentation du modem que les commandes AT sont les mêmes que celles dudit modem. Le cas échéant, le paramètre sera modifié en conséquence. Les configurations de chaque interface série sont habituellement automatiquement reconnues par le modem. Si la reconnaissance automatique n'est pas possible, il faut obligatoirement entrer manuellement les paramètres concernés. Les instructions se trouvent dans la documentation du modem. Les réglages se font généralement au moyen des commutateurs DIP ou à l'aide d'un programme terminal tel qu'Hyperterminal par exemple (composant de *Windows*).

**Modems recommandés par ELAU**

- Modem INSYS 56k (référence INSYS : 110.756.WD03)
- Modem INSYS 144 (référence INSYS : 110.6144.WD03)
- Modem Westermo TD-33 v. 90 (référence Westermo : 31790010)
- Modem Westermo TD-32 (référence Westermo : 3178-0040)

## 7.2.3 Paramètres du projet pour le PacDrive Controller MAx-4

- Pointez sur l'objet « Modem » dans la configuration de commande.

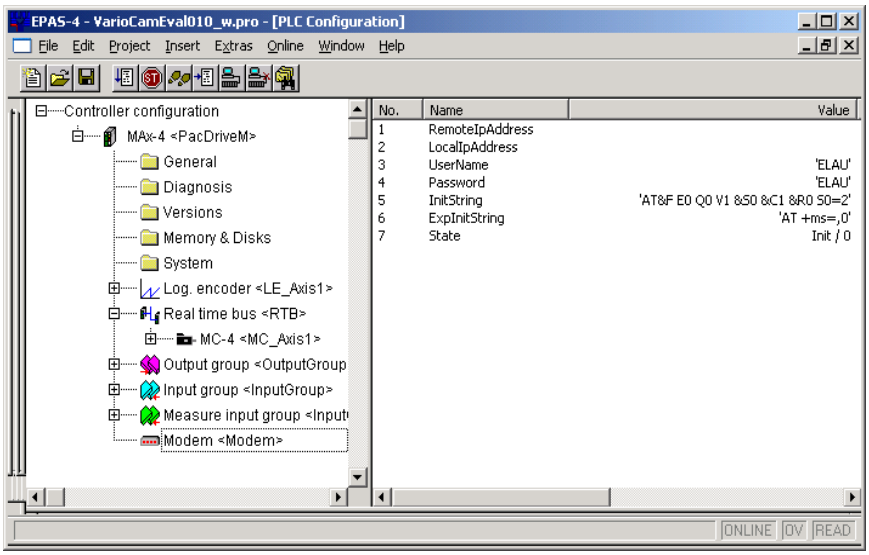



*Illustration 7-6: Objet « Modem » dans « PLC Configuration » (Configuration de commande)*

Les sept paramètres de l'objet modem ont la signification suivante :

### RemoteIpAddress

Il s'agit de l'adresse IP donnée à l'ordinateur qui est relié via le modem au PacDrive Controller MAx-4. Cette adresse IP est automatiquement attribuée à l'ordinateur lors de l'établissement de la liaison par le PacDrive Controller MAx-4. L'adresse par défaut est 190.200.100.101.

### LocalIpAddress

Il s'agit de l'adresse IP attribuée au MAx-4 lorsqu'il établit une liaison via un modem. Cette adresse IP est valable uniquement pour l'adaptateur PPP du MAx-4. L'adresse par défaut est 190.200.100.100 et ne peut pas être modifiée.

### UserName

Il s'agit du nom d'utilisateur qu'il faut entrer au moment du login de l'établissement de la connexion.

### Password

Il s'agit du mot de passe qu'il faut entrer lors du login de l'établissement de la connexion.

### InitString

La chaîne d'initialisation contient une suite de commandes AT qui sont envoyées au modem lors de l'initialisation. Le bon fonctionnement du modem du PacDrive Controller MAx-4 dépend de paramètres modem indispensables. Initstring par défaut configure ces paramètres et s'applique normalement à la plupart des modems classiques. Il sera bon toutefois de vérifier les commandes AT du modem dans la documentation du modem et de modifier l'Initstring le cas échéant afin d'activer les fonctions adéquates. Les configurations suivantes sont nécessaires :

InitString par défaut : AT&F E0 Q0 V1 &S0 &C1 &R0 S0=2

| Valeur | Signification |
|---|---|
| AT | Introduit une commande AT. |
| &F | Charge la configuration par défaut du modem. |
| E0 | Désactive la fonction « Echo des commandes » du modem. |
| Q0 | Active les messages. |
| V1 | Messages en mode verbal. |
| &S0 | Signal DSR : toujours 1 pour modem en attente. |
| &C1 | Active le signal DCD. (DCD reflète l'état de la porteuse.) |
| &R0 | Active le handshake matériel. (Le modem traite le signal CTS.) |
| S0=2 | Active la fonction Auto-Answer. (Le modem répond après la 2e sonnerie.) |

*Tableau 7-1: Signification de la chaîne d'initialisation InitString par défaut*

### ExpInitString

Du fait qu'*InitString* ne peut pas dépasser 40 caractères, et afin d'éviter d'éventuels problèmes avec le modem, il est possible avec *ExpInitString* d'entrer 40 signes supplémentaires qui sont envoyés au modem lors d'un second passage. La chaîne doit commencer par AT et ce, même s'il n'y a pas de commandes supplémentaires. *ExpInitString* est envoyé vers le modem uniquement si la chaîne InitString a été correctement traitée.

ExpInitString par défaut :  AT +MS=,0

| Valeur | Signification |
|---|---|
| AT | Introduit une commande AT. |

| Valeur | Signification |
|---|---|
| +MS=,0 | Désactive la reconnaissance automatique de la vitesse de transmission.<br>Cette commande sert uniquement lorsque des problèmes de handshake surviennent lors de la sélection du modem et il se peut qu'elle n'existe pas sur votre modem s'il n'est pas équipé d'une puce Rockwell.<br>Le modem recommandé supporte cette commande. |

*Tableau 7-2: Signification de la chaîne ExpInitString par défaut*

### Etat

Il s'agit de l'indication de l'état du modem.

| Valeur | Signification |
|---|---|
| Init / 0 | Le modem est contrôlé. |
| No Modem / 1 | Il n'y a pas de modem raccordé au port COM1 / Modem éteint. |
| Offline / 2 | Le modem est en mode Offline (pas d'appel). |
| Online / 3 | Le modem est en mode Online (appel entrant). |
| InitString OK / 4 | InitString accepté par le modem. |
| InitString ERROR / 5 | Impossible d'initialiser le modem avec InitString (la chaîne d'initialisation est peut-être erronée ou bien le modem n'a pas pu être configuré correctement). En réitérant la procédure d'initialisation, soit le modem doit être éteint et rallumé, soit le paramètre Com1User de l'objet MAx-4 doit être modifié puis écrit à nouveau via une liaison TCP/IP. |
| InitString TIMEOUT / 6 | Pas de réaction du modem à la chaîne InitString (cf. InitString ERROR / 5). |
| ExpInitString OK / 7 | Acceptation d'ExpInitString par le modem. |
| ExpInitString ERROR / 8 | L'InitString crée des erreurs au niveau du modem (cf. InitString ERROR / 5). |
| ExpInitString TIMEOUT / 9 | Pas de réponse du modem à la chaîne ExpInitString (cf InitString ERROR / 5). |

*Tableau 7-3: Etats possibles du modem*

### MAx-4.Com1User

Le paramètre *Com1User* de l'objet « MAx-4 » dans « General » a également une importance. C'est là qu'est configurée l'attribution du port série COM1 du PacDrive Controller MAx-4. Il faut régler sur « Modem / 1 » ou « Auto / 2 ».

Dans la configuration, « Modem / 1 » est toujours attribué au modem et le port COM1 est vérifié jusqu'à ce qu'un modem soit trouvé. Le contrôle a lieu toutes les 10 secondes.
Dans la configuration « Auto / 2 », le port COM1 est interrogé. Si un modem allumé y est détecté, celui-ci est initialisé ; sinon le port est toujours libéré pour l'EPAS-4.
Le paramètre n'est actif que si sa configuration a été modifiée dans le PacDrive Controller MAx-4. Pour réinitialiser un modem par exemple, le paramètre doit impérativement être réglé en passant de 'Modem' ou d''Auto' à 'No modem', et ensuite, il sera configuré selon l'état choisi. Lorsque l'état « Modem / 1 » ainsi que l'état « Auto / 2 » sont configurés, le PacDrive Controller MAx-4 réagit au débranchement ou au raccordement d'appareils.



**REMARQUE**

Le paramètre *Com1User* a une incidence directe sur le port COM1. Ce paramètre ne doit par conséquent être modifié ni par une liaison série EPAS-4 ni par une liaison modem. En effet, le port COM serait alors déconnecté pour la liaison et cela entraînerait un grave défaut de communication. Lors de la télémaintenance via un modem, il faut donc absolument que la liaison TCP/IP soit toujours fonctionnelle localement afin de pouvoir changer d'interface série le cas échéant.

### MAx-4.Com1UserState

Le paramètre Max4.Com1UserState renvoie l'état du port COM1.

| Valeur | Signification |
|---|---|
| Init / 0 | Vérification des appareils raccordés. |
| EPAS-4 / 1 | Com1User = EPAS-4 |
| Modem -> EPAS-4 / 2 | Com1User = modem. Pas de modem détecté ou impossible d'initialiser le modem. |
| Auto -> Search / 3 | Com1User = auto. Recherche d'un modem. |
| Modem -> Search / 4 | Com1User = modem. Recherche d'un modem. |
| Modem -> Test / 5 | Com1User = modem. Initialisation du modem détecté. |
| Modem -> Offline / 6 | Com1User = modem. Détection d'un modem et initialisation réussie. |
| Modem -> Online / 7 | Com1User = modem. Le modem reçoit un appel entrant. |

| Valeur | Signification |
|---|---|
| None / 8 | Com1User = non défini. |
| Auto -> Test / 9 | Com1User = auto. Initialisation d'un modem détecté. |
| Auto -> EPAS-4 / 10 | Com1User = auto. Pas de modem détecté ou bien le modem n'a pas pu être initialisé. |
| Auto -> Modem Offline / 11 | Com1User = auto. Détection d'un modem et initialisation réussie. |
| Auto -> Modem Online / 12 | Com1User = auto. Le modem reçoit un appel entrant. |

*Tableau 7-4: Etats possibles du MAx-4.Com1UserState*

## 7.2.4 Configuration de l’ordinateur distant

Pour relier un ordinateur avec le PacDrive Controller MAx-4 via un modem, les conditions suivantes doivent être remplies :

- EPAS-4 installé
- Réseau d’accès à distance installé (Windows 95/98) ou service d’accès distant RAS (Windows NT) avec TCP/IP
- Modem installé

L’installation de l’atelier logiciel EPAS-4 est décrite dans le manuel d’utilisation d’EPAS-4. L’installation d’une connexion à distance et d’un modem est présentée ci-dessous pour Windows 95/98 et pour Windows NT.

### Windows 95/98

Installation du réseau d’accès à distance

- Il convient de vérifier que le réseau d’accès à distance est déjà installé : dans le panneau de configuration, sélectionnez « Add/Remove Programs » (Ajout/Suppression de programmes).

*Illustration 7-7: Control Panel (Panneau de configuration) Win 95/98*

PDM_UserMan_Teleservice_fr.fm

- Dans l'onglet « Windows Setup » (Installation de Windows), sélectionnez « Communications » ; cliquez ensuite sur le bouton « Details... » .

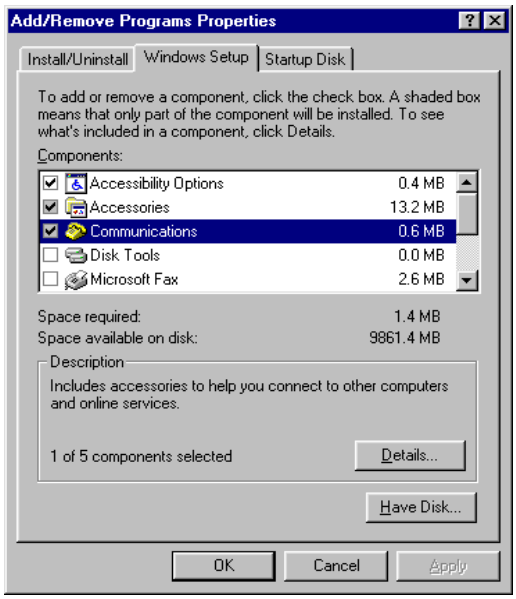

Add/Remove Programs Properties
Install/Uninstall
Windows Setup
Startup Disk
To add or remove a component, click the check box. A shaded box means that only part of the component will be installed. To see what's included in a component, click Details.
Components:
Accessibility Options 0.4 MB
Accessories 13.2 MB
Communications 0.6 MB
Disk Tools 0.0 MB
Microsoft Fax 2.6 MB
Space required: 1.4 MB
Space available on disk: 9861.4 MB
Description
Includes accessories to help you connect to other computers and online services.
1 of 5 components selected
Details...
Have Disk...
OK
Cancel
Apply

*Illustration 7-8: Add/Remove Programs Properties / Windows Setup (Propriétés de Ajout/Suppression Programmes / Installation de Windows)*

- Sélectionnez « Dial-Up Networking » (Accès Réseau à Distance), s'il n'est pas déjà marqué.

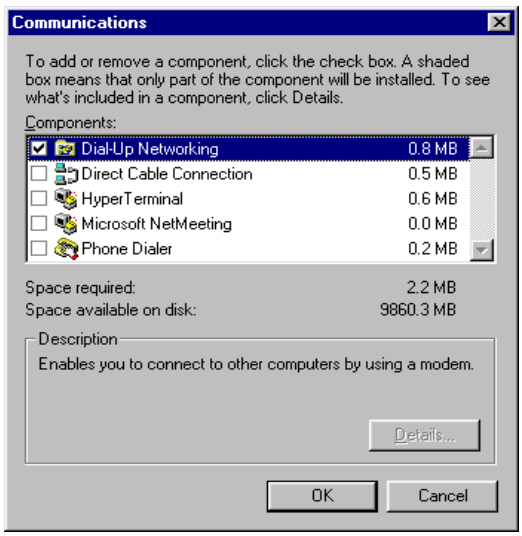

Communications
To add or remove a component, click the check box. A shaded box means that only part of the component will be installed. To see what's included in a component, click Details.
Components:
Dial-Up Networking 0.8 MB
Direct Cable Connection 0.5 MB
HyperTerminal 0.6 MB
Microsoft NetMeeting 0.0 MB
Phone Dialer 0.2 MB
Space required: 2.2 MB
Space available on disk: 9860.3 MB
Description
Enables you to connect to other computers by using a modem.
Details...
OK
Cancel

*Illustration 7-9: Communications*

- Validez avec « OK ».

Avec cette validation, le réseau d'accès à distance va être installé. Pour cela, il faut avoir le CD d'installation de Windows. Après redémarrage de l'ordinateur, le réseau d'accès à distance est disponible. Du fait que le protocole TCP/IP est mis en place

PDM_UserMan_Teleservice_fr.fm

ensuite, le redémarrage n'est pas nécessaire dans l'immédiat. En effet, après installation dudit protocole, il faut de toute façon redémarrer l'ordinateur.

Installation du protocole TCP/IP

- Si le protocole TCP/IP n'est pas installé sur votre ordinateur, effectuez l'installation maintenant. La procédure d'installation est décrite dans le manuel d'utilisation d'EPAS-4.

Configuration de la connexion à distance et du modem

- Lancez le réseau d'accès à distance (pour Win95 = Menu Start (Démarrer) -> Programs (Programmes) -> Accessories (Accessoires) -> Dial Up Networking (Accès Réseau à Distance) ; pour Win98 = Menu Start (Démarrer) -> Programs (Programmes) -> Accessories (Accessoires) -> Communications (Communications) -> Dial Up Networking (Accès Réseau à Distance). Lors du premier appel du programme ou lors de la sélection de Make New Connection (Nouvelle connexion), un assistant s'affiche pour aider à la configuration de la connexion. Si aucun modem n'est configuré, la procédure d'installation de modem est exécutée.

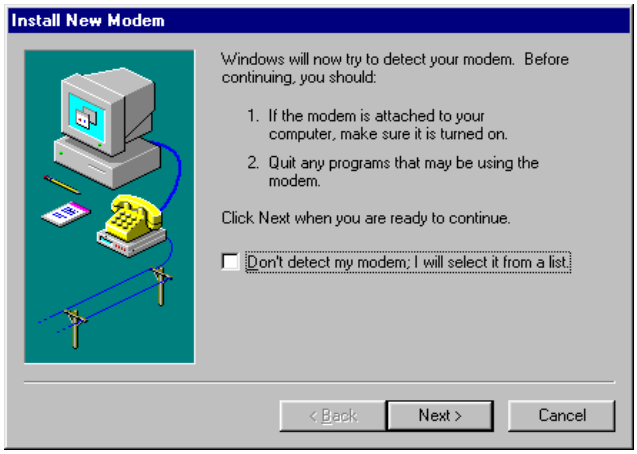

Install New Modem
Windows will now try to detect your modem. Before continuing, you should:
1. If the modem is attached to your computer, make sure it is turned on.
2. Quit any programs that may be using the modem.
Click Next when you are ready to continue.
Don't detect my modem; I will select it from a list.
< Back
Next >
Cancel

*Illustration 7-10: Boîte de dialogue Install New Modem (installation nouveau modem)*

Il est possible, à ce niveau, d'avoir la reconnaissance automatique du modem ou bien de choisir un modem à partir d'une liste.

- Validez avec « Next » (Suivant) afin de poursuivre la procédure d'installation.

**REMARQUE**

En cas de problèmes lors de l'installation, consultez le manuel d'utilisation de votre modem.

- Entrez les paramètres de localisation du modem.

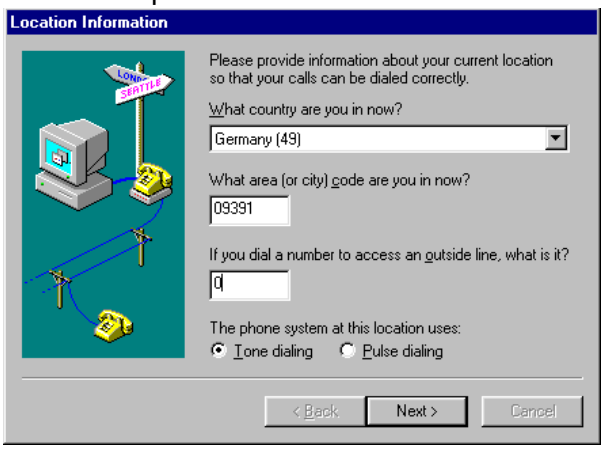

Location Information
Please provide information about your current location so that your calls can be dialed correctly.
What country are you in now?
Germany (49)
What area (or city) code are you in now?
09391
If you dial a number to access an outside line, what is it?
0
The phone system at this location uses:
Tone dialing
Pulse dialing
< Back
Next >
Cancel

*Illustration 7-11: Location Information (paramètre local du modem)*

Le modem est maintenant installé et de nouvelles connexions entre l'ordinateur et le PacDrive Controller MAx-4 peuvent être créées avec « Make New Connection » (Nouvelle connexion).

Création d'une connexion entre l'ordinateur et le PacDrive Controller MAx-4

- Sélectionnez « Make New Connection » (Nouvelle connexion) dans l'accès réseau à distance « Dial-Up Networking » (Accès Réseau à Distance).

*Illustration 7-12: Dial-Up Networking (Make New Connection)*

- Entrez un nom pour la connexion et sélectionnez le modem. Validez avec le bouton « Configure... » (Configurer) pour procéder à la configuration du modem.

PDM_UserMan_Teleservice_fr.fm

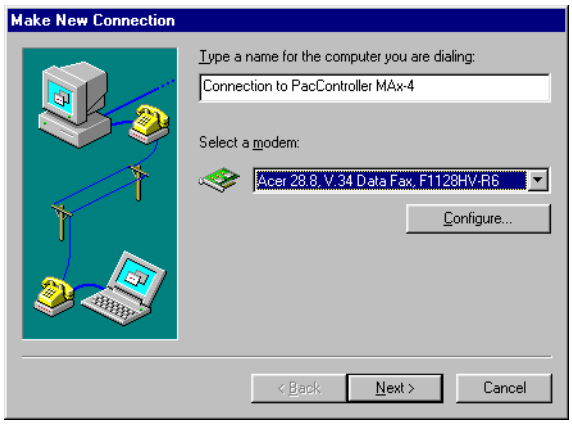



*Illustration 7-13: Make New Connection (Nouvelle connexion)*

- Sélectionnez l'onglet « Settings » (Paramètres). Vérifiez les paramètres suivants :
  - Bit de données : 8
  - Parité : aucune
  - Bits d'arrêt :1
- Dans la fenêtre « Advanced Connection Settings » (Paramètres de connexion avancés), cochez les cases *Use error control (Utiliser le contrôle d'erreurs)* et *Compress data (Compresser les données)* ainsi que *Use flow control (Utiliser le contrôle de flux)* et Hardware (RTS / CTS) (Matériel).

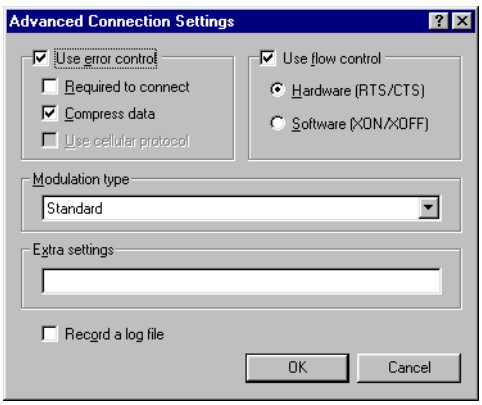



*Illustration 7-14: Advanced Connection Settings*

- Validez deux fois avec OK et tapez sur « Next » (Suivant) pour terminer la première phase de l'Assistant.

- Dans la deuxième phase, entrez le numéro d'appel du PacDrive Controller MAx-4 et validez en cliquant sur « Next ».

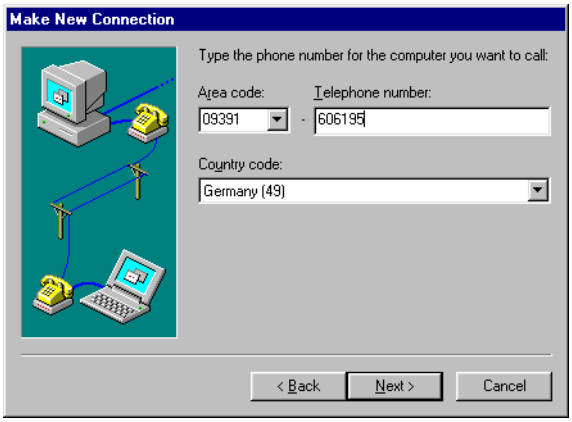



*Illustration 7-15: Make New Connection (Nouvelle connexion)*

- La connexion est ensuite créée en cliquant sur « Finish » (Terminer).
- Appelez la fenêtre de dialogue des propriétés avec un clic droit de la souris et sélectionnez dans le menu contextuel « Properties ».

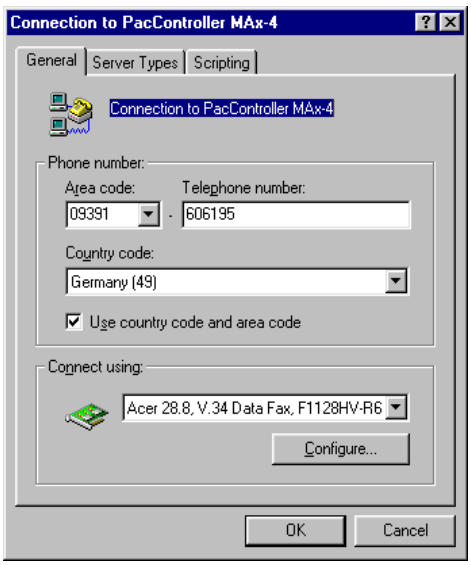



*Illustration 7-16: Connection to PacDrive Controller MAx-4 / General*

- Sélectionnez « Server Types » (Types de serveur) et paramétrez comme suit :
  - Type of Dial-Up Server: (Type de serveur d'accès distant) PPP: Internet, Windows NT Server, Windows 95 ou Windows 98

– Désactivez les options avancées
– Activez uniquement TCP/IP comme protocole réseau autorisé

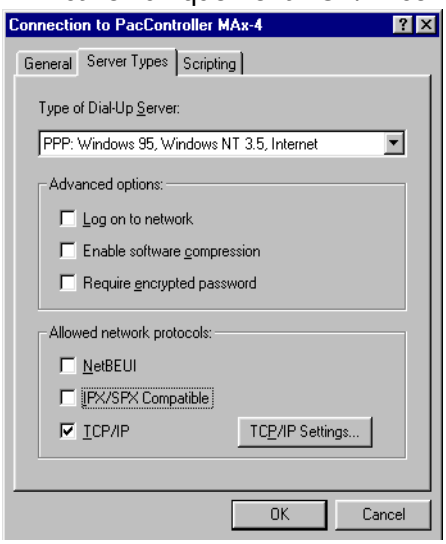



*Illustration 7-17: Connection to PacDrive Controller MAx-4 / Server Types*

- Entrez les paramètres TCP/IP à l'aide des boutons correspondants.
  – Cochez « Server assigned IP address » (Adresse IP attribuée par serveur)
  – Cochez « Server assigned name server addresses » (Adresses de serveurs de noms attribuées par serveur)
  – Cochez « Use IP header compression » (Utiliser compression d'en-tête)
  – Activez « Use default gateway on remote network » (Utiliser la passerelle par défaut pour le réseau distant)

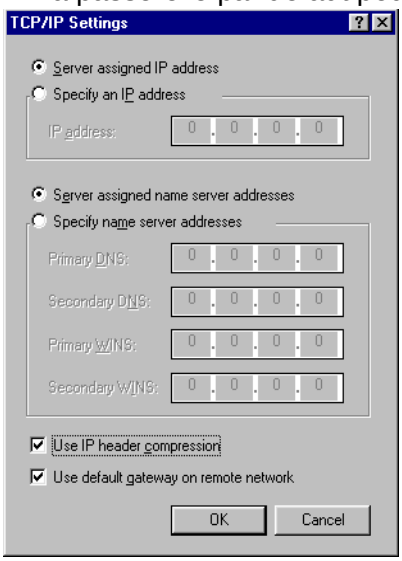



*Illustration 7-18: TCP / IP Settings (Paramètres TCP / IP)*

PDM_UserMan_Teleservice_fr.fm

- Cliquez sur OK.
- La connexion est maintenant disponible.

**Activez « Use default gateway on remote network » (Utiliser la passerelle par défaut pour le réseau distant)**

Ce paramètre indique si la passerelle par défaut est utilisée pour le réseau distant pour la durée de la connexion. Lorsque que vous créez un accès réseau à distance ou un réseau VPN, une nouvelle route par défaut est ajoutée à la table de routage IP, utilisant la connexion au serveur RAS (service d'accès distant). Cette route par défaut reçoit la plus petite métrique. Au cas où une route par défaut existe déjà, elle reste dans la table de routage, mais une métrique plus élevée lui est attribuée. Le fait d'ajouter une nouvelle route par défaut avec une métrique plus petite peut empêcher l'accès aux emplacements auxquels la première route par défaut accédait.

Un ordinateur ayant ce système d'exploitation dans un réseau d'entreprise a par ex.  une connexion LAN dans ce réseau et il est configuré avec une passerelle par défaut d'un routeur IP local. Avec l'option **« Use default gateway on remote network »** (Utiliser la passerelle par défaut pour le réseau à distance), la passerelle par défaut devient le fournisseur d'accès internet (FAI), et non le routeur local de l'entreprise, lorsqu'un utilisateur sélectionne sur cet ordinateur un FAI pour accéder à l'internet. C'est pourquoi aucun emplacement du réseau d'entreprise, à l'exception de ceux du réseau LAN, n'est accessible pendant la durée de la connexion avec le FAI. Si l'option **« Use default gateway on remote network »** (Utiliser la passerelle par défaut pour le réseau à distance) n'est pas sélectionnée, la première route par défaut reste inchangée ; il n'y a pas d'ajout de nouvelle route par défaut et les emplacements dans Internet ne sont pas accessibles.

**Windows NT 4.0**

Installation du protocole TCP / IP

- Si le protocole TCP/IP n'est pas installé sur votre ordinateur, effectuez l'installation maintenant. La procédure d'installation est décrite dans le Mode d'emploi EPAS-4.

Installation du service d'accès distant RAS (Remote Access Service)

- Sélectionnez dans le panneau de configuration « Network » (Réseau).

*Illustration 7-19: Win NT Control Panel (Panneau de configuration)*

- Dans la boîte de dialogue « Network » (Réseau), dans l'onglet « Services », vérifiez que le service d'accès distant existe. Dans le cas contraire, il faut l'installer avec le bouton « Add » (Ajouter).

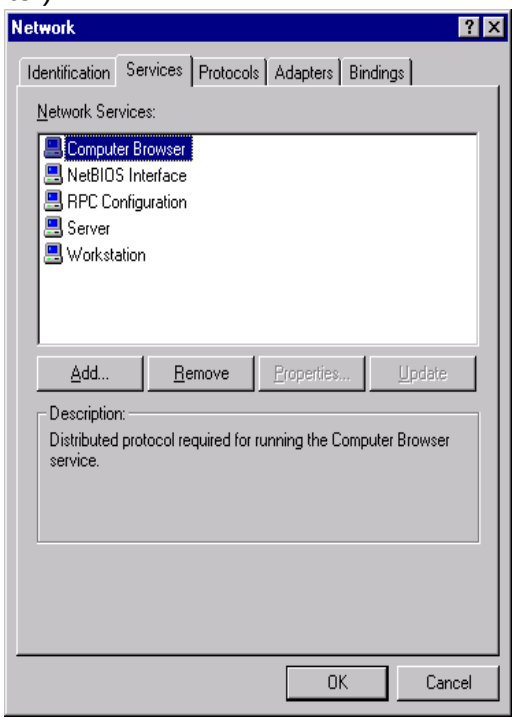



*Illustration 7-20: Win NT Control Panel / Network / Services (Panneau de configuration Win NT / Réseau / Services)*

- Sélectionnez le service d’accès distant et validez avec « OK ».

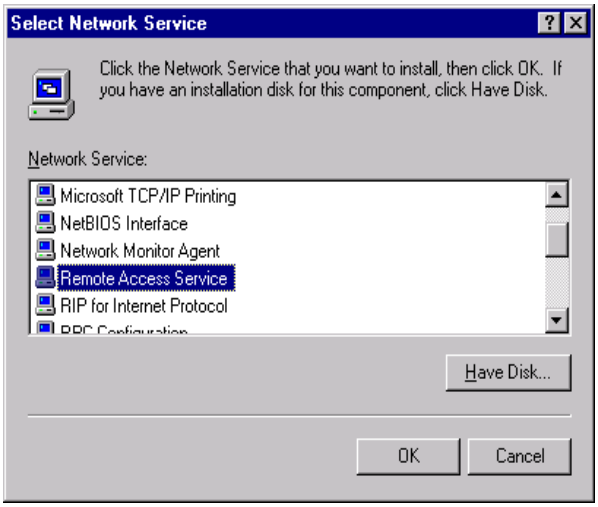

Select Network Service
Click the Network Service that you want to install, then click OK. If you have an installation disk for this component, click Have Disk.
Network Service:
Microsoft TCP/IP Printing
NetBIOS Interface
Network Monitor Agent
Remote Access Service
RIP for Internet Protocol
Have Disk...
OK
Cancel

*Illustration 7-21: Win NT Control Panel / Network / Remote Access Service (Panneau de configuration Win NT / Réseau / Service d'accès distant)*

- Si aucun modem n’est installé, la fenêtre RAS Setup (Installation du service d’accès distant) demande si le programme d’installation du modem doit être appelé. Confirmez en cliquant sur
« Yes » (Oui).

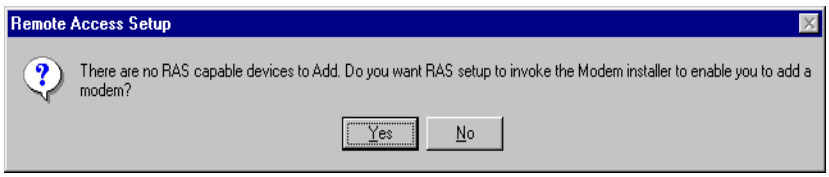

Remote Access Setup
There are no RAS capable devices to Add. Do you want RAS setup to invoke the Modem installer to enable you to add a modem?
Yes
No

*Illustration 7-22: RAS Setup (Installation du service d’accès distant)*

Lors de l’installation du modem, il est possible d’opter pour la détection automatique du modem ou pour la sélection dans une liste.

- Cliquez sur le bouton « Next » (Suivant) afin que la procédure d’installation se poursuive.

**REMARQUE**

En cas de problèmes lors de l’installation, consultez le manuel d’utilisation de votre modem.

PDM_UserMan_Teleservice_fr.fm

- Maintenant, le Service RAS détecte et sélectionne le modem (validez avec OK).

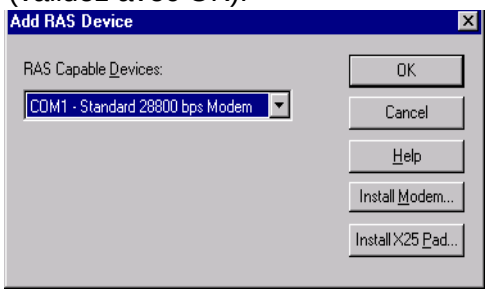



*Illustration 7-23: Add RAS Device (Ajout d'un périphérique RAS)*

- Dans la fenêtre d'installation du service d'accès distant, sélectionnez un périphérique et confirmez avec le bouton « Network… » (Réseau).

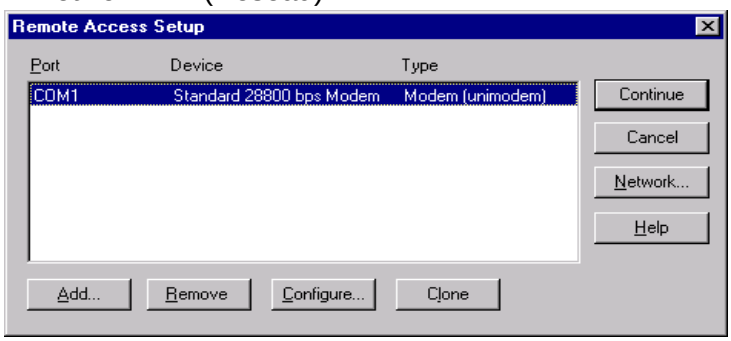



*Illustration 7-24: RAS Setup (Installation du service d'accès distant)*

- Cochez uniquement le protocole client TCP/IP et validez avec « OK ».

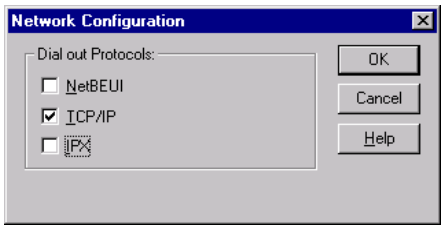



*Illustration 7-25: Network Configuration ()*

- Activez « Configure... » et vérifiez que la case « Dial out only » (Appels sortants seulement) est cochée. (Si l'ordinateur reçoit

PDM_UserMan_Teleservice_fr.fm

les appels pour d'autres applications, la case « Dial out and Receive calls » (Appels sortants et entrants) sera cochée.)

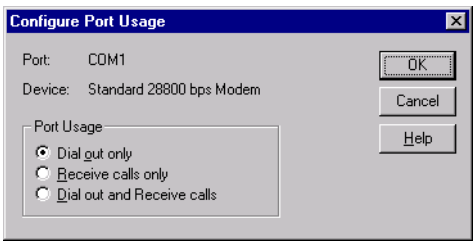



*Illustration 7-26: Configure Port Usage (Configuration de l'utilisation du port)*

- Validez avec « OK ».
- Cliquez « Next » pour terminer la configuration.

Le service d'accès distant est maintenant disponible.

Création d'une connexion entre l'ordinateur et le PacDrive Controller MAx-4

- Lancez le réseau d'accès à distance avec Start/Programs/ Accessories/Dial Up-Networking (Accès réseau à distance). Une nouvelle entrée d'annuaire est requise ; validez avec OK.

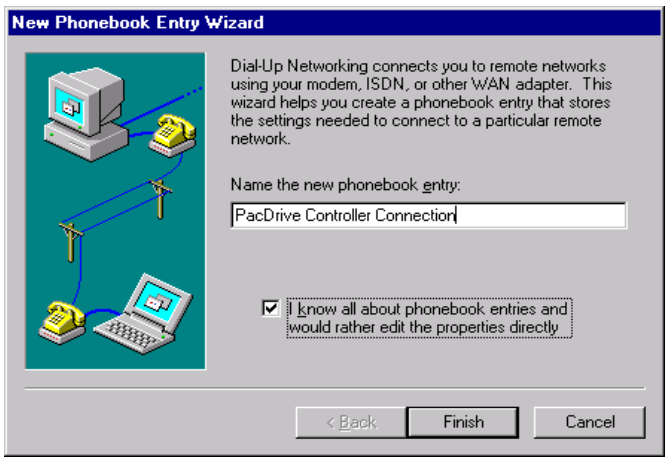

New Phonebook Entry Wizard
Dial-Up Networking connects you to remote networks using your modem, ISDN, or other WAN adapter. This wizard helps you create a phonebook entry that stores the settings needed to connect to a particular remote network.
Name the new phonebook entry:
PacDrive Controller Connection
I know all about phonebook entries and would rather edit the properties directly
< Back
Finish
Cancel

*Illustration 7-27: New Directory Entry (Assistant de nouvelle entrée d'annuaire)*

- Entrez un nom pour la connexion et cochez le champ « No further information required » . Si vous connaissez déjà tout des entrées d'annuaire, vous pouvez passer à la modification des propriétés directement et validez avec « Finish » (Terminer). La boîte de dialogue de configuration d'une nouvelle entrée d'annuaire s'affiche.

- Dans l’onglet « Entry », tapez le numéro à appeler pour le PacDrive Controller MAx-4.

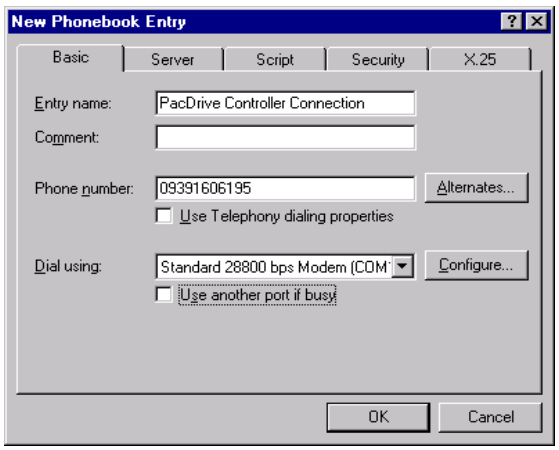



*Illustration 7-28: New Directory Entry*

- Cliquez sur « Configure... » pour ouvrir la boîte de dialogue de la configuration du modem et activez tous les paramètres matériel.

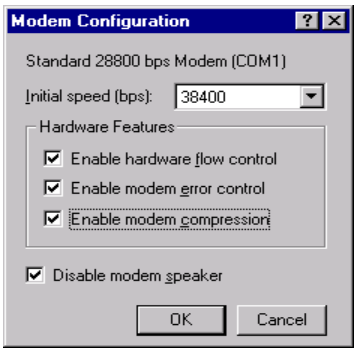



*Illustration 7-29: Modem Configuration*

- Validez avec « OK ».
- Dans l’onglet « Server » , entrez les paramètres suivants :
  - Type of Dial-Up Server: (Type de serveur d’accès distant) PPP: Windows NT, Windows 95 ou 98, Internet
  - Activez seulement le protocole réseau TCP/IP
  - Désactivez « Softwarekomprimierung » (compression logicielle)
  - Activez « LCP-Erweiterungen für PPP aktivieren » (Activer

extensions LCP pour PPP)

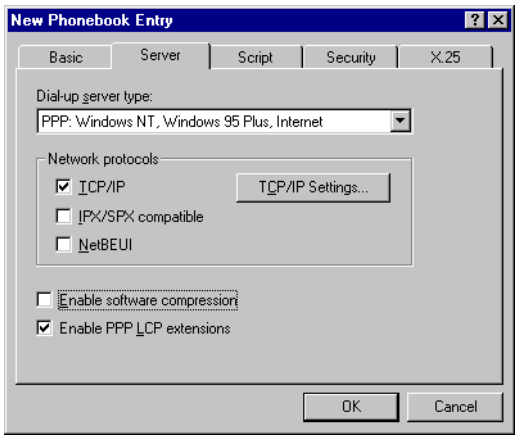



*Illustration 7-30: New Directory Entry / Server*

- Les paramètres TCP/IP sont validés en cochant les cases choisies.
  - Cochez « Server assigned IP address » (Adresse IP attribuée par serveur)
  - Cochez « Server assigned name server addresses » (Adresses de serveurs de noms attribuées par serveur)
  - Cochez « Use IP header compression » (Utiliser compression d'en-tête)
  - Activez « Use default gateway on remote network » (Utiliser passerelle par défaut pour le réseau distant)

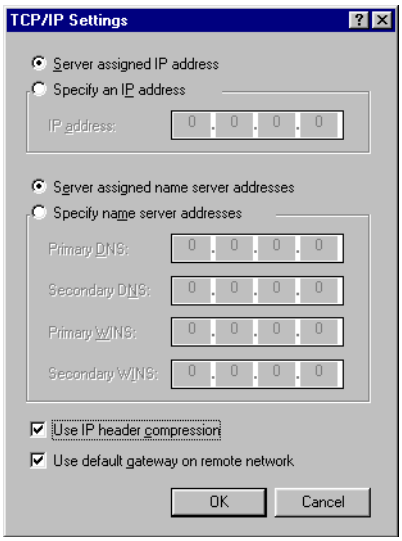



*Illustration 7-31: TCP/IP Settings (Configuration TCP/IP)*

- Validez avec « OK ».

- Dans l'onglet « Security » (Sécurité), il faut choisir impérativement le paramètre « Accept any authentification including clear text/...(even non-encoded) » (Accepter toute authentification y compris du texte vide).

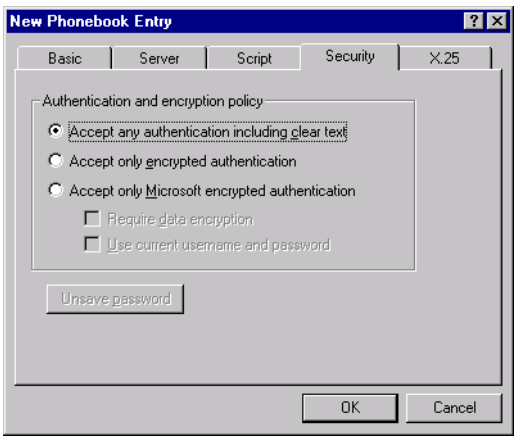



*Illustration 7-32: New Directory Entry / Security*

- Validez avec « OK ».

La connexion est maintenant disponible pour la sélection.

### Windows 2000

Installation du protocole TCP/IP

- Si le protocole TCP/IP n'est pas installé sur votre ordinateur, effectuez l'installation maintenant. La procédure d'installation est décrite dans le Mode d'emploi EPAS-4.

Installation du modem sur l'ordinateur

- Ouvrez le panneau de configuration (Start/Settings/Control Panel).
- Sélectionnez « Phone and Modem Options » (Options de modem et téléphonie).

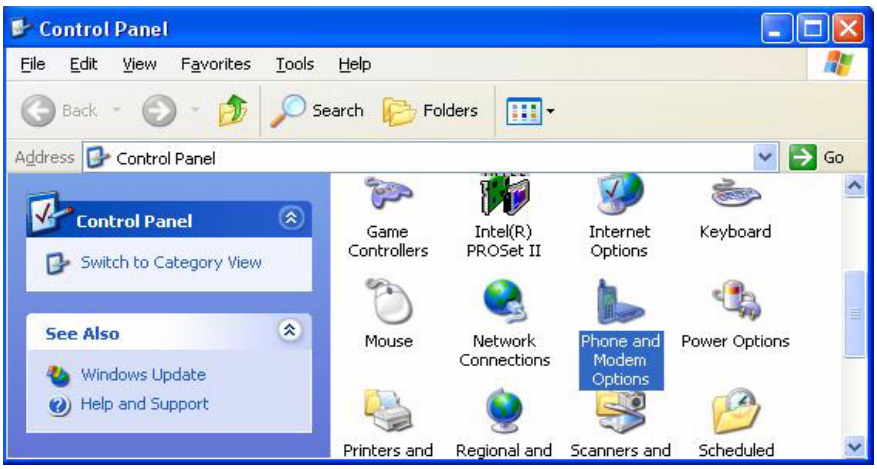



*Illustration 7-33: Control Panel (Panneau de configuration) Windows 2000/XP*

PDM_UserMan_Teleservice_fr.fm

- Entrez l'information concernant votre emplacement (*Location Information)*.

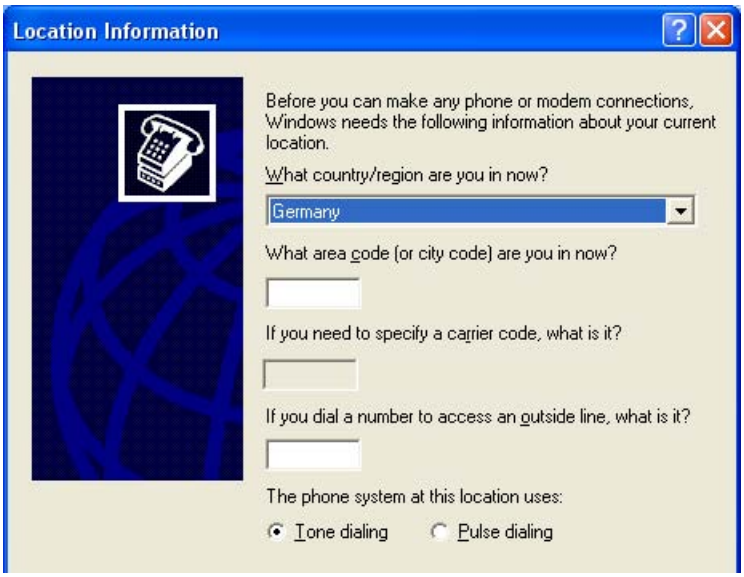

Location Information
Before you can make any phone or modem connections, Windows needs the following information about your current location.
What country/region are you in now?
Germany
What area code (or city code) are you in now?
If you need to specify a carrier code, what is it?
If you dial a number to access an outside line, what is it?
The phone system at this location uses:
Tone dialing
Pulse dialing

*Illustration 7-34: Boîte de dialogue « Location Information »*

- Dans l'onglet Modems, sélectionnez « Add to... » (Ajouter)

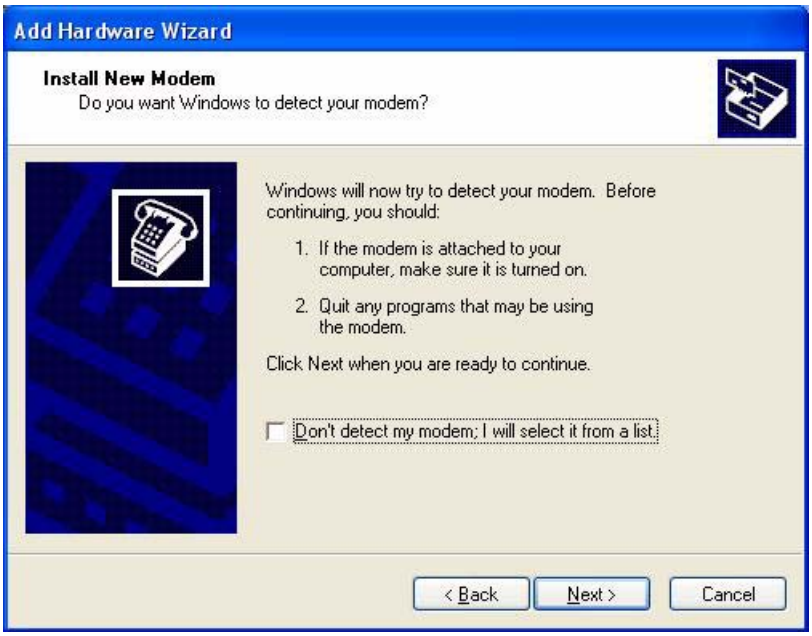

Add Hardware Wizard
Install New Modem
Do you want Windows to detect your modem?
Windows will now try to detect your modem. Before continuing, you should:
1. If the modem is attached to your computer, make sure it is turned on.
2. Quit any programs that may be using the modem.
Click Next when you are ready to continue.
Don't detect my modem; I will select it from a list.
< Back
Next >
Cancel

*Illustration 7-35: Add Hardware Wizard (Assistant Ajout de matériel) / Win 2000/ XP*

- Suivez les instructions de l'Assistant Ajout de matériel.

PDM_UserMan_Teleservice_fr.fm

**REMARQUE**

En cas de problèmes lors de l'installation, consultez le mode d'emploi de votre modem.

Création d'une connexion vers le PacDrive Controller MAx-4

- Lancez le réseau d'accès à distance (Start -> Programs -> Accessories -> Communications -> Network Connections).
- Dans la fenêtre Network Connection (Connexions Réseau), sélectionnez « Create a new connection » (Etablir une nouvelle connexion).

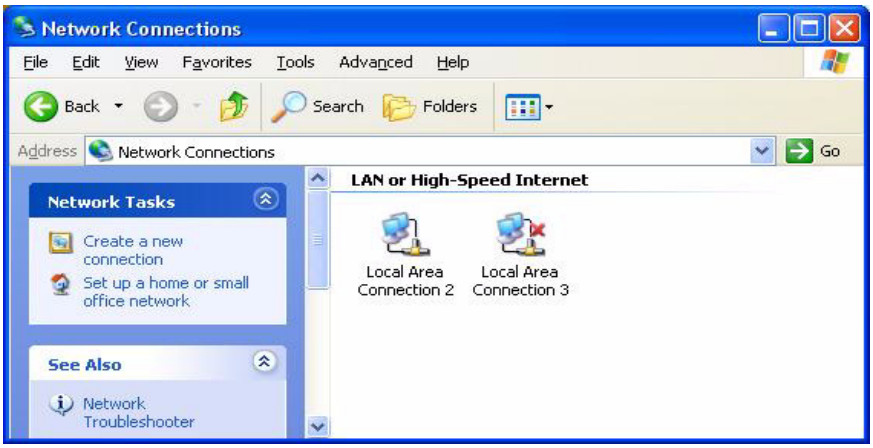



*Illustration 7-36: Network Connections / Create a new connection (Connexions Réseau/Etablir une nouvelle connexion)*

New Connection Wizard
Network Connection Type
What do you want to do?
Connect to the Internet
Connect to the Internet so you can browse the Web and read email.
Connect to the network at my workplace
Connect to a business network (using dial-up or VPN) so you can work from home, a field office, or another location.
Set up a home or small office network
Connect to an existing home or small office network or set up a new one.
Set up an advanced connection
Connect directly to another computer using your serial, parallel, or infrared port, or set up this computer so that other computers can connect to it.
< Back
Next >
Cancel

*Illustration 7-37: New Connection Wizard (Assistant Connexion Réseau) / 1*

New Connection Wizard
Network Connection
How do you want to connect to the network at your workplace?
Create the following connection:
Dial-up connection
Connect using a modem and a regular phone line or an Integrated Services Digital Network (ISDN) phone line.
Virtual Private Network connection
Connect to the network using a virtual private network (VPN) connection over the Internet.
< Back
Next >
Cancel

*Illustration 7-38: New Connection Wizard (Assistant Connexion Réseau) / 2*

New Connection Wizard
Connection Name
Specify a name for this connection to your workplace.
Type a name for this connection in the following box.
Company Name
For example, you could type the name of your workplace or the name of a server you will connect to.
< Back
Next >
Cancel

*Illustration 7-39: New Connection Wizard (Assistant Connexion Réseau) / 3*

New Connection Wizard
Phone Number to Dial
What is the phone number you will use to make this connection?
Type the phone number below.
Phone number:
11
You might need to include a "1" or the area code, or both. If you are not sure you need the extra numbers, dial the phone number on your telephone. If you hear a modem sound, the number dialed is correct.
< Back
Next >
Cancel

*Illustration 7-40: New Connection Wizard (Assistant Connexion Réseau) / 4*

Une fois la connexion créée, elle se trouve dans le répertoire *Network Connections* (Connexions Réseau) et peut être démarrée à cet endroit.

New Connection Wizard
Completing the New Connection Wizard
You have successfully completed the steps needed to create the following connection:
Dial-up Connection
• Share with all users of this computer
The connection will be saved in the Network Connections folder.
Add a shortcut to this connection to my desktop
To create the connection and close this wizard, click Finish.
< Back
Finish
Cancel

*Illustration 7-41: Vérifier la connexion ou lancer l'établissement de la connexion*

*Illustration 7-42: Boîte de dialogue pour établir la connexion*

- Pour vérifier les paramètres, sélectionnez « Properties » (Propriétés).
- Dans l'onglet « General », cliquez le bouton « Configure... »

PDM_UserMan_Teleservice_fr.fm

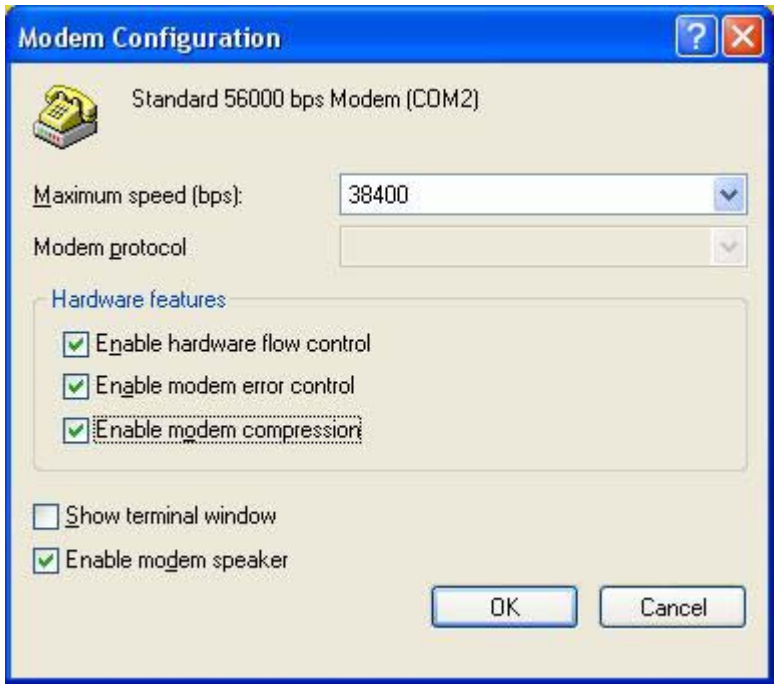



*Illustration 7-43: Modem Configuration*

- Sélectionnez les paramètres matériels *Enable hardware flow control* (Activer le contrôle de flux), *Enable modem error control* (Activer le contrôle d'erreur du modem) et *Enable modem compression* (Activer la compression du modem).
- Validez avec « OK ».
- Cliquez sur l'onglet « Networking » (Réseau).

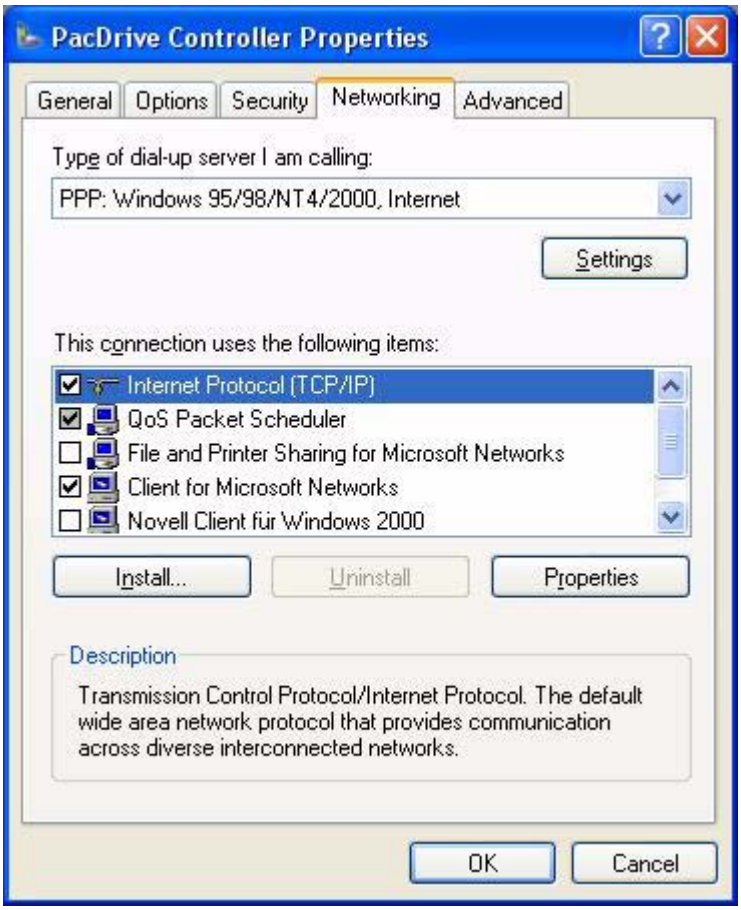



*Illustration 7-44: Propriétés de la connexion / Réseau*

- Vérifiez le type du serveur d'accès à distance appelé (« Type of dial-up server I am calling »). Il faut obligatoirement sélectionner : *PPP:Windows 95/98/NT4/2000, Internet*.
- Vérifiez que seul le protocole *Internet Protocol (TCP/IP)* est coché.
- Cochez *Internet Protocol (TCP/IP)*.
- Cliquez sur le bouton « Properties » et sélectionnez
  - *Obtain an IP address automatically (Obtenir une adresse IP automatiquement)*
  - *Obtain DNS server address automatically (Obtenir les adresses des serveurs DNS automatiquement)*

Internet Protocol (TCP/IP) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address:
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server:
Alternate DNS server:
Advanced...
OK
Cancel

*Illustration 7-45: Internet Protocol (TCP/IP) Properties (Propriétés du protocole Internet (TCP/IP))*

- Cliquez deux fois sur « OK ».

*Illustration 7-46: Boîte de dialogue pour établir la connexion*

- Entrez le nom d'utilisateur (voir le paramètre *UserName*) et le mot de passe (voir le paramètre *Password*) pour établir la connexion.
- Etablissez la connexion avec le bouton « Dial » (Se connecter).
- La connexion est maintenant disponible.

### 7.2.5 Etablissement de la connexion entre le PacDrive Controller et l'ordinateur distant

La procédure de connexion au PacDrive Controller MAx-4 est décrite dans le détail dans le Mode d'emploi de l'atelier logiciel Automation Toolkit EPAS-4. Les points suivants sont traités :

- Tests de la liaison TCP/IP
- Installation du serveur passerelle (gateway)
- Installation d'un canal de communication dans EPAS-4

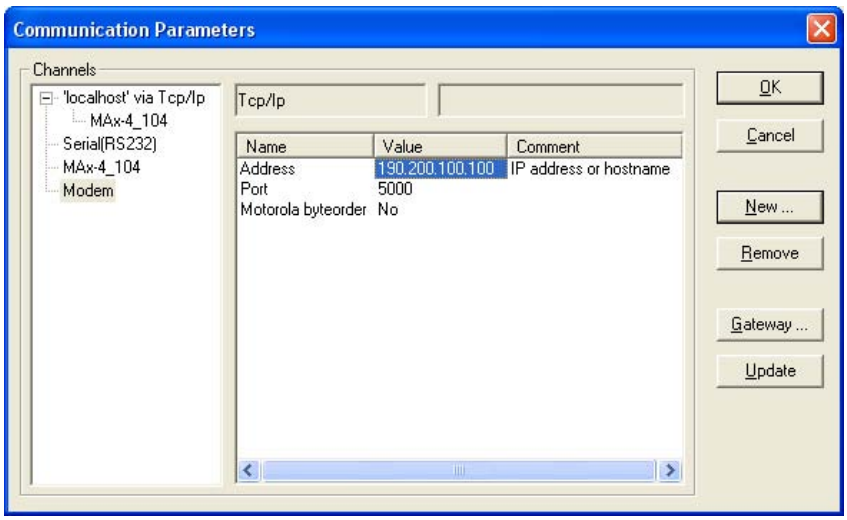



*Illustration 7-47: « Communication Parameters » : Paramètres de communication pendant la connexion du modem*

## 7.2.6 Recherche/localisation de défauts

**REMARQUE**

Il est toujours recommandé de procéder à un test et de lire attentivement le manuel avant la mise en service.

| **Le modem numérote, mais le modem du PacDrive Controller MAx-4 n'accepte pas l'appel.** | |
|---|---|
| Le modem du PacDrive Controller MAx-4 n'est pas encore ou n'est pas correctement initialisé. Si dans PLC CONFIGURATION \| MODEM \| le paramètre *Status* indique « offline », le modem est initialisé. Assurez-vous que l'InitString configure le modem pour accepter les appels. | |
| Cause 1 :<br><br>Solution : | Le paramètre *Com1User* dans EPAS-4 \| PLC CONFIGURATION \| GENERAL n'est pas correct.<br>Vérifiez le paramètre *Com1User*. Le paramètre doit être sur « Modem » ou « Auto ». |
| Cause 2 :<br><br><br>Solution : | La chaîne d'initialisation n'est pas correcte. Dans ce cas, après initialisation du modem, le message « [Exp]InitString ERROR » est affiché dans le paramètre *Status*.<br>Vérifiez la chaîne d'initialisation dans le paramètre [*Exp*]*InitString* dans la configuration de commande. |
| Cause 3 :<br><br><br>Solution : | Le modem n'est pas branché correctement ou mal initialisé. Dans ce cas, le message « [Exp]InitString TIMEOUT » est affiché dans le paramètre *Status*.<br>Vérifiez l'alimentation et le câble série ainsi que le paramètre [*Exp*]*InitString*. |
| Cause 4 :<br>Solution : | Le paramètre *ExpInitString* est erroné.<br>Vérifiez le paramètre *ExpInitString*. |
| Cause 5 :<br><br><br>Solution : | Le jeu de composants de votre modem n'est pas un chip Rockwell. Dans ce cas, il se peut que la commande de reconnaissance de vitesse de connexion ne soit pas reconnue correctement.<br>Utilisez les commandes s'appliquant à votre modem. Elles sont indiquées dans la documentation du modem.<br>Si vous n'avez pas de problème de contrôle de flux (handshake) entre vos modems, vous pouvez remplacer le contenu du paramètre *ExpInitString* par la commande « AT ». |
| Cause 6 :<br><br>Solution : | Le câble série ou le câble de téléphone ne sont pas adaptés au modem du PacDrive Controller MAx-4.<br>Utilisez le câble livré par votre fournisseur. |

PDM_UserMan_Teleservice_fr.fm

| **Le modem raccroche avant que la connexion ne soit établie.** | |
|---|---|
| | |
| Cause 1 :<br><br>Solution : | Le temps d'attente du modem avant d'interrompre l'établissement de la connexion est trop court.<br>WIN9x : Dial-Up Network (Réseau d'accès à distance) -> Properties -> Configuration -> Settings : décochez « Interrupt Dialing » (Interrompre la numérotation).<br>WIN NT : Control Panel -> Modem -> Properties -> Settings : décochez « Interrupt Dialing » (Interrompre la numérotation). |

| **Le modem décroche, mais le handshake n'est pas établi ou est interrompu.** | |
|---|---|
| | |
| Cause 1 :<br><br><br>Solution : | Les modems utilisent divers paramètres de vitesse. Un modem est configuré avec « Connect with 19200 bps » (Connexion à 19200 bps) tandis qu'un autre ne supporte que 14400 bps.<br>Configurez les modems avec la sélection automatique de vitesse.<br>WIN9x : Dial-Up Networking (Accès réseau à distance) -> Properties (Propriétés) -> Connect via (Connecter via) -> Bouton « Configure », onglet « General » : Décochez « Only connect at this speed » (Se connecter uniquement à cette vitesse).<br>WIN NT : Control Panel -> Modems -> Properties: Décochez « Only connect at this speed » (Se connecter uniquement à cette vitesse).<br>Le cas échéant, réglez correctement les interrupteurs DIP de votre modem. |
| Cause 2 :<br><br>Solution : | Les modems ne reconnaissent pas leurs protocoles réciproques.<br>Désactivez l'ajustement automatique de la modulation. Conservez le paramètre *ExpInitString* de la configuration commande « +MS=,0 ». Si votre modem ne supporte pas cette commande, consultez le manuel du modem. |
| Cause 3 :<br>Solution : | La liaison téléphonique est trop mauvaise.<br>Une solution consiste à changer de réseau téléphonique ou de branchement. Appelez le modem du MAx-4 avec le téléphone. Attendez le signal handshake du modem (commence généralement avec un sifflement). Si le signal est plus ou moins brouillé, avec des bruits forts de claquement ou de crépitements sur la ligne, cela signifie que la qualité de la ligne est trop mauvaise pour une liaison. |

| **La liaison est établie, mais l'accès réseau à distance affiche « Verifying user name and password... » (Vérification du nom d'utilisateur et du mot de passe...). Et la communication est coupée.** | |
|---|---|
| | |
| Cause 1 :<br><br>Solution : | Le mot de passe ou le nom d'utilisateur ne correspondent pas à ceux mis en mémoire dans le PacDrive Controller MAx-4 (Paramètre *UserName* et *Password*) .<br>Entrez le bon mot de passe et le nom d'utilisateur dans EPAS-4. |
| Cause 2 :<br><br>Solution : | Pour WinNT uniquement : L'accès réseau à distance a envoyé un mot de passe crypté. MAx-4 ne le reconnaît pas.<br>Activez l'accès réseau à distance (Dial Up Networking) ->cliquez sur « More » (Plus) -> « Entry and Modem Properties... » -> Security (Sécurité) -> « Accept any authentification.. » (Accepter toute authentification...) |
| Cause 3 :<br>Solution : | Les propriétés du réseau d'accès à distance ne sont pas configurées correctement.<br>Win9x : dans « Dial Up Networking » (Accès réseau à distance) -> bouton « Properties » (Propriétés) -> onglet « Server types » (Types de serveur) : décochez tout sauf TCP/IP ; Type of Dial-Up Server: (Type de serveur d'accès distant) = Internet, WIN NT Server, WIN9x<br>WIN NT : Dial Up Networking (Accès réseau à distance) -> bouton « Next » (Suivant) -> « Edit Entry and Modem Properties » (Modifier l'entrée et les paramètres du modem) -> onglet « Server » (Serveur) -> Type of Dial-Up Server: (Type de serveur d'accès distant) = PPP, Internet, WIN NT Server, WIN9x , cochez TCP/IP et les extensions LCP, décochez tous les autres. |
| Cause 4 :<br><br>Solution : | Le protocole TCP/IP n'a pas pu être lancé car les adresses IP sont erronées (utilisation d'adresses IP réservées, par exemple).<br>Utilisez les adresses IP correctes. |
| Cause 5 :<br><br>Solution : | Le paramètre *RemoteIpAddress* de la configuration de commande (PLC Configuration) / Modem ne correspond pas à la valeur entrée dans le réseau d'accès à distance.<br>Vérifiez les paramètres sous<br>WIN9x : Dial-Up Networking (Accès réseau à distance) -> bouton « Properties » -> onglet « Server types » -> bouton « TCP/IP Settings ».<br>WIN NT : Dial-Up Networking (Accès réseau à distance) -> bouton « More » -> « Edit Entry and Modem Properties » -> onglet « Server » -> bouton « TCP/IP Settings ». |

PDM_UserMan_Teleservice_fr.fm

| **La liaison est établie, mais l'accès réseau à distance affiche « Verifying user name and password... » (Vérification du nom d'utilisateur et du mot de passe...). Et la communication est coupée.** | |
|---|---|
| | |
| Cause 6 :<br>Solution : | Les paramètres des modems sont différents.<br>Entrez « 8 data bits, no parity, 1 stop bit » (8 bits de données, pas de parité, 1 bit d'arrêt).<br>WIN9x : Dial-Up Networking (Accès réseau à distance) -> Properties -> onglet « General » -> bouton « Configure »-> onglet « Settings » (Paramètres) : « 8 data bits, no parity, 1 stop bit » (8 bits de données, pas de parité, 1 bit d'arrêt).<br>WIN NT : Control Panel (Panneau de configuration) -> Modems -> Properties -> onglet « Settings » : « 8 data bits, no parity, 1 stop bit » (8 bits de données, pas de parité, 1 bit d'arrêt).<br>Dans le même onglet (Illustration 7-14) -> bouton « Advanced Connection Settings » : activez Use Flow Control : « Hardware ( RTS / CTS ) » et « Error control » : Data Compression.<br>Si votre modem est équipé de commutateurs DIP, ils seront réglés le cas échéant. |

| **EPAS-4 affiche le message : « Communication Error: Logging out. » lors de l'essai d'ouverture de session (login).** | |
|---|---|
| | |
| Cause 1 :<br><br><br>Solution : | Les paramètres de communication de l'EPAS-4, menu ONLINE \| COMMUNICATION PARAMETERS, ne correspondent pas avec l'adresse IP locale *LocalIpAddress*. *LocalIpAddress* est 190.200.100.100 par défaut.<br>Vérifiez les paramètres de communication. |
| Cause 2 :<br><br>Solution : | La liaison modem est trop lente, en dessous de 9600 bps par exemple. EPAS-4 ne détecte pas de liaison car aucun paquet complet de données n'est transmis dans la limite du time-out. Augmentez la valeur de *EPAS_StandardTimeout* dans EPAS-4.INI sous le répertoire WINDOWS. Redémarrez EPAS-4. |
| Cause 3 :<br><br>Solution : | Il existe déjà dans un réseau local un appareil ayant l'adresse IP attribuée au PacDrive Controller MAx-4.<br>Modifiez l'adresse IP du PacDrive Controller MAx-4. Isolez du réseau l'ordinateur distant. Pour les questions concernant vos adresses IP, adressez-vous à votre administrateur système. |
| Cause 4 :<br>Solution : | Problème de réseau dont la cause est inconnue.<br>Isolez du réseau l'ordinateur distant et faites une réinstallation complète du réseau d'accès à distance. |

| **EPAS-4 interrompt la connexion en cours : « Communication Error : Logging out ».** | |
|---|---|
| | |
| Cause 1 :<br><br><br><br>Solution : | La liaison modem n'est pas interrompue :<br>La limite de time-out de la liaison modem a été dépassée. Cela vient de la capacité trop réduite des liaisons lentes et par là, des temps trop longs de transmission de paquets de données.<br>Augmentez dans EPAS-4.INI sous le répertoire WINDOWS la valeur de **EPAS_StandardTimeout** en entrant « 10 ou plus ». Il faut aussi éventuellement augmenter la valeur de **EPAS_ExtendedTimeout**. Cette valeur sera d'autant plus élevée que la qualité de la ligne est mauvaise et que le modem est lent. |
| Cause 2 :<br><br><br>Solution : | Le modem raccroche :<br>La connexion est interrompue à cause de la mauvaise qualité de la ligne, par exemple.<br>Installez la liaison sur une ligne de meilleure qualité. |
| Cause 3 :<br>Solution : | Le PacDrive Controller MAx-4 a été redémarré.<br>Attendez quelques secondes que la procédure de démarrage soit terminée et numérotez à nouveau. |
| Cause 4 :<br><br><br>Solution : | La connexion a été coupée par Windows car il n'y avait plus de données envoyées par l'adaptateur d'accès à distance pendant un certain temps.<br>WIN9x : Dial-Up Network (Réseau d'accès à distance) -> Properties -> Configuration -> Settings : décochez « Disconnect after idle time ».<br>WIN NT : Control Panel -> Modem -> Properties -> Settings : décochez « Disconnect after idle time ». |
| Cause 5 :<br><br><br><br>Solution : | Une information venant d'un appareil extérieur (impulsion de comptage, appel en attente, etc. ) est envoyée au modem. Le modem a alors coupé la connexion.<br>Désactivez les fonctions complémentaires de ce genre avant d'établir une connexion. Consultez le manuel de votre installation téléphonique. |

PDM_UserMan_Teleservice_fr.fm

## 7.3 Connexion à distance au PacDrive Controller via l’ordinateur distant

### 7.3.1 Principe de fonctionnement

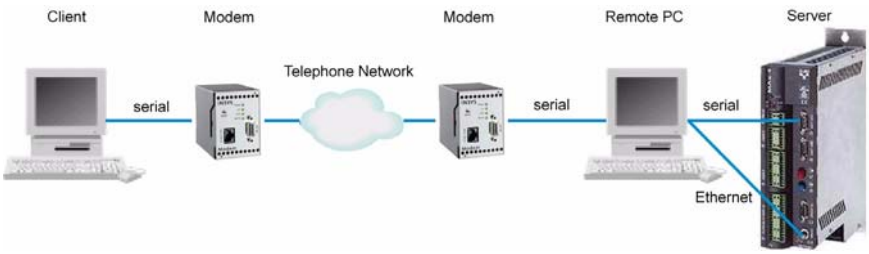



*Illustration 7-48: Accès direct au PacDrive Controller MAx-4 via l’ordinateur distant*

**Avantages**

- Il est possible de se connecter à plusieurs PacDrive Controller MAx-4
- Il est possible d’installer la fonctionnalité « Call back » (rappel téléphonique) de Windows
- Mise en oeuvre d’une connexion au bureau à distance (remote desk connection) et d’outils de commande à distance (Remote Control Tools) tels que le VNC (virtual network computing)

**Inconvénients**

- Configuration de système plus lourde par rapport à une connexion directe avec le PacDrive Controller MAx-4
- Il est nécessaire d’avoir un ordinateur distant avec le PacDrive Controller MAx-4

## 7.4 Connexion à distance avec plusieurs automates PacDrive Controller via le routeur

### 7.4.1 Principe de fonctionnement

*Illustration 7-49: Connexion à distance directe via le routeur vers plusieurs automates PacDrive Controller MAx-4*

**Avantages**

- Il est possible de se connecter à plusieurs PacDrive Controller MAx-4

**Inconvénients**

- Configuration de système plus lourde par rapport à une connexion directe avec un PacDrive Controller MAx-4

PDM_UserMan_Teleservice_fr.fm

# 7.5 Connexion via Internet

## 7.5.1 Connexion directe via une adresse IP

Pour pouvoir accepter une connexion via Internet, il faut obligatoirement connaître l'adresse IP du PacDrive Controller MAx-4 et de l'ordinateur distant. TCP/IP sera installé côté serveur et côté client.

A noter que l'adresse IP ne peut pas être choisie arbitrairement. Une adresse univoque pour le réseau mondial sera attribuée.

Cette adresse est attribuée par un organisme central international, le NIC (Network Information Center). Vous trouverez de plus amples renseignements sur le site www.denic.de.

**Avantages**

- Simplicité de configuration

**Inconvénients**

- Demander une adresse univoque pour un PacDrive Controller MAx-4 ne s'avère pas réaliste (en raison des coûts, du nombre limité d'adresses IP).

Mais si un ordinateur distant se connecte à Internet via un fournisseur d'accès internet, cet ordinateur se voit attribuer automatiquement de manière dynamique une adresse IP. Par conséquent, si cette adresse IP est connue, il est possible créer une connexion avec cet ordinateur distant.

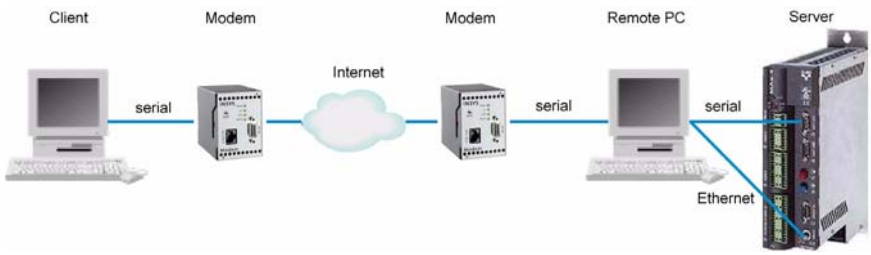



*Illustration 7-50: Connexion directe via une adresse IP et l'ordinateur distant*

**Avantages**

- Il est possible de se connecter à plusieurs PacDrive Controller MAx-4
- Simplicité de configuration

**Inconvénients**

- L'adresse IP change à chaque connexion.

## 7.5.2 Connexion VPN (Virtual Private Network)

Il est possible de créer une connexion réseau à l’aide d’une connexion VPN via une connexion Internet.

Cela s’applique par exemple à un PacDrive Controller MAx-4 relié à un réseau intranet qui dispose d’un accès Internet.

**Avantage**

- Cette connexion est semblable à celle d’un réseau local (LAN).

**Inconvénient**

- La liaison est complexe quant à sa configuration.

# 7.6 Accès au PacDrive Controller via un navigateur classique (non disponible)

## 7.6.1 Principe de fonctionnement

La télémaintenance via Internet, avec un navigateur « classique », ne donne à l'utilisateur qu'un accès très limité aux fonctions système PacDrive™.

**AVERTISSEMENT !**

Surveillance réduite auprès de la machine !

Risques de dommages corporels et matériels !

- La télémaintenance sera effectuée uniquement s'il y a un opérateur disponible auprès de la machine et si cet opérateur est qualifié.
- Il doit y avoir un contact permanent (par téléphone ou vidéo-conférence) avec l'opérateur.

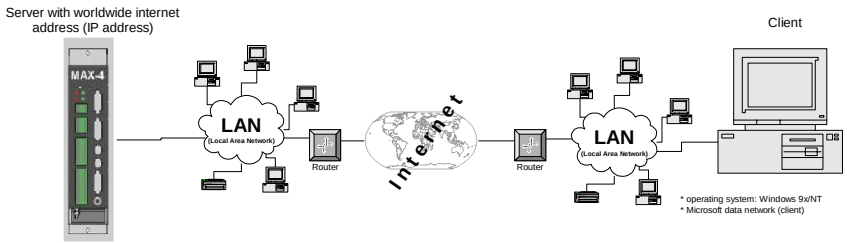



*Illustration 7-51: Connexion Internet*

### Connexions possibles

- Réseau local (LAN)
- Réseau d'accès à distance via modem
- Connexion à l'Internet

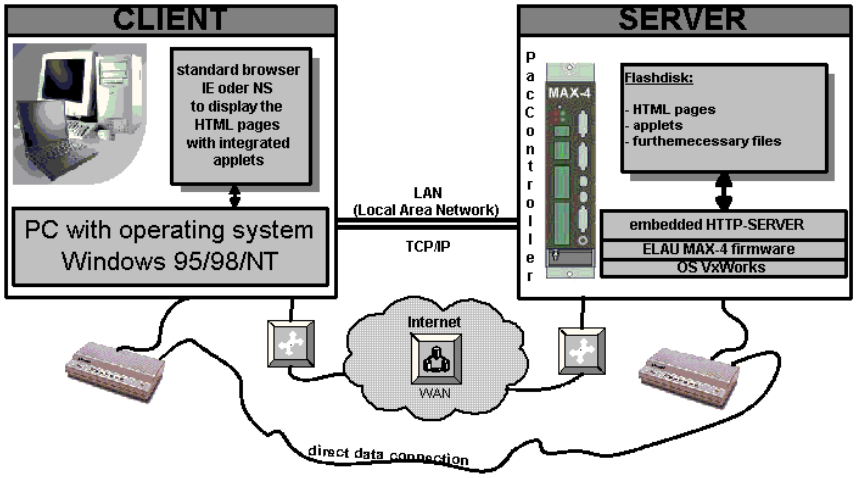



*Illustration 7-52: Composants du télédiagnostic*

Le serveur HTTP relié au PacDrive Controller MAx-4 permet de télécharger des pages HTML où les applettes de diagnostic sont intégrées. Les applettes sont implémentées sur une page avec divers onglets HTML. Cela garantit une facilité de maniement du diagnostic ainsi qu'une configuration individuelle de celui-ci. Il est possible de définir soi-même les éléments de la mise en forme, le contenu et la présentation d'une page HTLM. La page ainsi créée peut être envoyée à l'aide d'un client FTP sur le disque Flash du PacDrive Controller MAx-4.

# 8 Réseaux

Ce chapitre a pour but de vous familiariser avec le monde des réseaux et de l'interconnexion entre les divers équipements.

Vous apprendrez à connaître les finalités, les avantages, les possibilités de mise en œuvre ainsi que les technologies inhérentes à ce domaine. En maîtrisant les supports de transmission (encore appelés « médias ») et les protocoles de communication, y compris les fondements du protocole TCP/IP, vous saurez détecter et résoudre les problèmes de gestion de ce protocole.

## 8.1 Concept de base

L'environnement réseau de Microsoft fait la distinction entre l'environnement peer-to-peer (pair à pair) ou groupe de travail (pas de serveur dédié) et les domaines (à base de serveur).

### 8.1.1 Peer to Peer

Tous les ordinateurs sont égaux et forment un groupe de travail.

Chaque ordinateur met à la disposition des autres des ressources et lui-même a accès à leurs ressources. Pour obtenir l'accès, l'utilisateur devra posséder un compte utilisateur sur chaque ordinateur.

**Avantages** Pas de frais supplémentaires pour le serveur et/ou le système d'exploitation

**Inconvénients** Pas d'administration utilisateur centralisée. A partir d'une taille de réseau plus importante, la coordination devient extrêmement lourde à gérer.

Le peer-to-peer est une solution peu onéreuse pour des petits réseaux (dix postes ou moins) qui n'ont pas pour priorité la sécurité.

**Le système d'exploitation peer-to-peer**

Les systèmes d'exploitation peer-to-peer sont en fait des systèmes pour des ordinateurs autonomes (stand alone) auxquels une fonction réseau a été ajoutée. (Win95, Win98, WinME).

Il manque toutefois à ces systèmes les critères déterminants inhérents aux « véritables » systèmes d'exploitation réseau comme, par exemple, l'authentification au moment de l'ouverture de session. Concrètement, le peer-to-peer est limité aux réseaux très petits qui sont organisés en groupes de travail composés d'ordinateurs déjà autonomes.

Dans Windows XP, il faut également une procédure d'ouverture de session.

## 8.1.2 Client-Serveur

Le concept client/serveur met pratiquement toujours en oeuvre une administration utilisateur centralisée. Les utilisateurs sont répartis en divers groupes ; à ces groupes sont attribués des droits précis.

Le concept client-serveur signifie : répartition des tâches. Un ou plusieurs ordinateurs mettent à disposition, en tant que serveurs, des ressources et des services centralisés. Tous les autres ordinateurs peuvent y accéder en tant que clients après une ouverture de session réussie.

En présence de réseaux plus importants, les divers services sont habituellement répartis sur plusieurs serveurs spécialisés.

Les services types de serveur sont : serveur de fichiers (file server), serveur d'imprimante (print server), serveur mail, serveur web, serveur proxy.

**Avantages**

Les nouveaux utilisateurs sont rapidement intégrés grâce à l'administration utilisateur centralisée.

L'utilisateur peut entrer en communication avec le réseau sous son nom et ce, à partir de n'importe quel ordinateur, et accéder aux ressources dont il a besoin.

**Inconvénients**

Le coût est plus élevé s'il faut ajouter des ordinateurs ou des systèmes d'exploitation.

**Système d'exploitation client-serveur**

Ces systèmes d'exploitation mettent à disposition les services centralisés auxquels peuvent alors accéder plusieurs clients, comme le serveur Win2000.

**Résumé**

Puisque les réseaux peer-to-peer existants peuvent être intégrés dans les réseaux client-serveur, les petites et moyennes entreprises choisiront de commencer avec un réseau peer-to-peer et plus tard, en cas de besoin, d'étendre le réseau avec une structure client-serveur. Les deux concepts sont compatibles et peuvent être combinés si nécessaire.

# 8.2 Topologies

## 8.2.1 Les topologies

### Distinction entre les topologies logique et physique

La mise en réseau crée un trafic de données et comme pour tout type de trafic, lors de sa mise en oeuvre, il convient de faire la distinction entre les voies de communication et les règles de circulation. Cela s'applique aussi à l'informatique.

### Topologie physique

La topologie physique d'un réseau concerne les voies de communication ; c'est l'architecture physique du réseau. Il s'agit, pour simplifier, de la manière dont le câblage est réalisé pour établir une communication avec fil.

Les topologies physiques de base sont :

- Topologie en bus
- Topologie en étoile
- Topologie en anneau

Dans la réalité, il y a combinaison des trois topologies.

### Topologie logique

La topologie logique d'un réseau correspond aux règles de circulation de base appliquées aux voies de communication. Il s'agit en outre de définir qui a le droit d'accéder au support ou média de transmission.

### Combinaison des topologies

Dans la pratique, les deux concepts de topologie sont étroitement liés et normalement une topologie physique donnée va s'allier à une topologie logique donnée. Les topologies physiques et logiques ne sont pas forcément identiques.

## 8.2.2 Topologie en bus

### Tous sur le même câble !

La topologie en bus se caractérise par un câble unique appelé « bus ». Tous les équipements sont reliés à ce bus et doivent partager ce support (shared media).

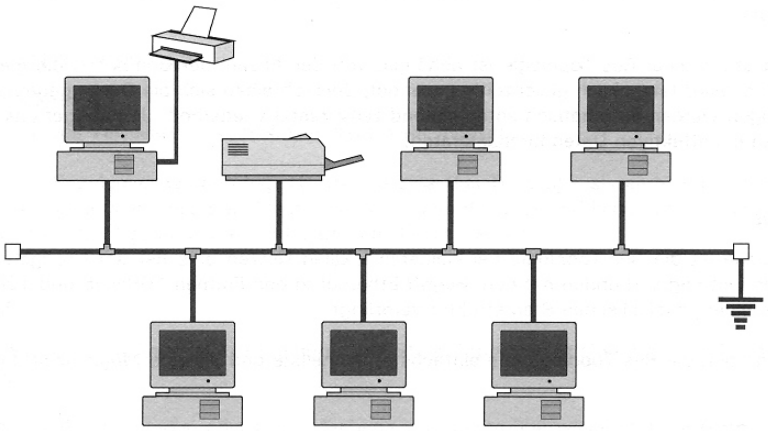


*Illustration 8-1: Topologie en bus*

**Avantages de la topologie en bus**

- Coût relativement peu élevé car il faut très peu de câbles.
- Si une station est en panne, le reste du réseau n'est pas perturbé.

**Inconvénients de la topologie en bus**

- Toutes les données sont transmises via un câble unique (conducteur).
- Il peut être assez facile d'espionner les transmissions de données.
- Un problème survenant au niveau du média de transmission et à un seul endroit du bus (câble défectueux, connecteur mal branché, carte réseau défectueuse) va affecter l'ensemble du réseau.
- La recherche de l'origine de la panne demande souvent beaucoup de moyens.
- Circulation de données importante du fait qu'une station envoie un message à toutes les autres.

## 8.2.3 Topologie en étoile

**A chacun son câble !**

Dans une topologie en étoile, chaque station est reliée, par un câble qui lui est propre, à un répartiteur central appelé hub ou concentrateur.

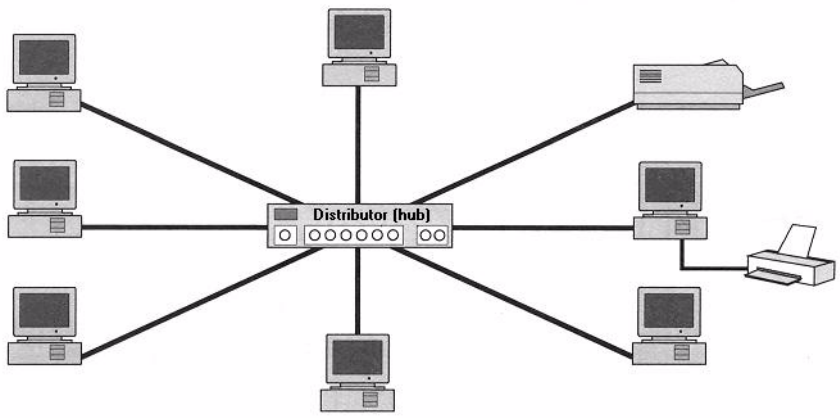



*Illustration 8-2: Topologie en étoile*

**Avantages de la topologie en étoile**

- Si une station est défaillante ou si un câble est défectueux, il n'y a pas d'incidence sur le reste du réseau.
- Les répartiteurs actifs jouent également un rôle d'amplificateur de signal.
- Si la fonctionnalité du répartiteur est adéquate, les ordinateurs ne partagent jamais leurs câbles respectifs pour la circulation des données. Cela signifie qu'ils s'approprient toute la bande passante, ce qui permet à cette topologie physique d'avoir des taux de débit plus élevés.
- Il est relativement facile d'ajouter d'autres stations et/ou d'autres répartiteurs.

**Inconvénients de la topologie en étoile**

- Câblage important
- Lors d'une panne du répartiteur, le réseau est paralysé.

## 8.2.4 Topologie en anneau

**Jamais sans ses voisins !**

Dans une topologie en anneau, le câblage forme une boucle fermée. Il n'y a ni début ni fin. Toutes les stations sont des éléments de l'anneau, elles traitent et amplifient les signaux qui arrivent sur le câble et les renvoient.

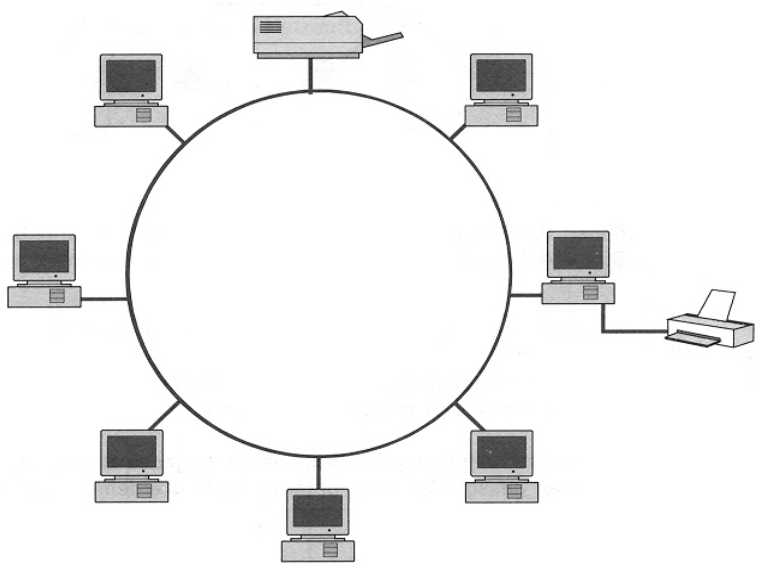


*Illustration 8-3: Topologie en anneau*

Cette topologie physique est rarement utilisée pour créer un réseau car il faut une quantité considérable de câbles.

Le principal inconvénient est celui de la défaillance d'une station ou d'une partie de câble, entraînant la paralysie de tout le réseau.

## 8.2.5 Topologies hydrides

### Combinaisons des topologies bus/étoile/anneau

Il est tout à fait possible de développer des topologies mixtes à partir des trois types précédents. En effet, la taille des réseaux d'entreprise augmente et, avec elle, l'utilisation probable de tous les types de topologies. De plus, la combinaison des topologies permet d'édifier un « backbone », une épine dorsale soutenant le reste du réseau.

**Backbone** Le « backbone » est la connexion physique de plusieurs réseaux, par exemple la liaison entre divers bâtiments et leurs réseaux respectifs.

### Topologie hydride bus / étoile

C'est une configuration courante où les topologies en étoile et en bus sont combinées.

Différents hubs constituent le coeur de l'étoile. Ces mêmes hubs sont connectés entre eux via un bus.

Un exemple simple va permettre d'illustrer cette architecture. Dans un immeuble de bureau à trois étages, chaque étage a son réseau en étoile. Les trois étages, plus exactement les hubs sont reliés entre eux via un câble unique (bus).

PDM_UserMan_Netze_fr.fm

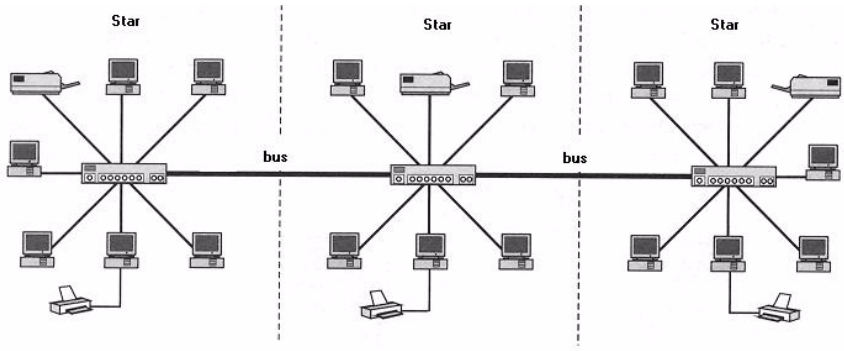



*Illustration 8-4: Topologie hybride étoile/bus*

En cas de rupture du câble du bus, les étages ne peuvent plus communiquer. En cas de communication des étages entre eux et à l'étage concerné est interrompue.

**Topologie étoile/étoile**

Chaque réseau en étoile possède un hub en son centre ; les hubs des divers réseaux en étoile sont reliés à un hub central.

Un autre exemple simple : dans un immeuble de bureau de trois étages, chaque étage est configuré en étoile. Les trois étages, donc les hubs des réseaux respectifs, sont reliés chacun par un câble à un hub central.

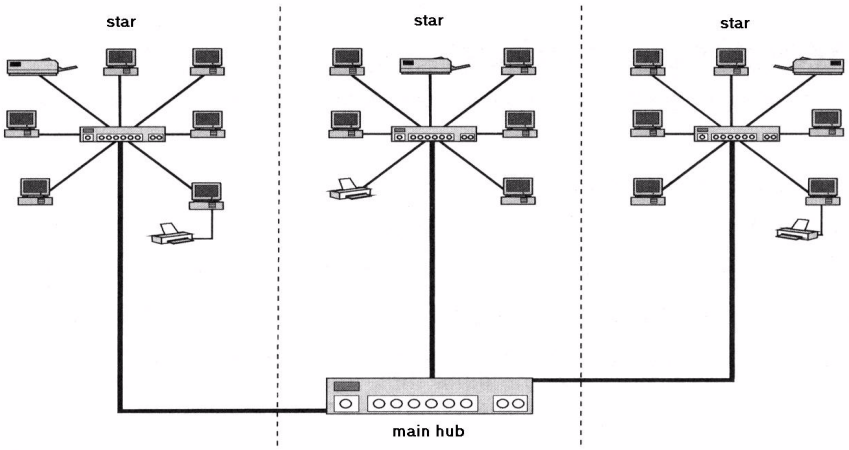



*Illustration 8-5: Topologie étoile/étoile*

Si le hub central est défaillant, la communication est encore possible au niveau de chaque étage. Si un câble du hub central relié à un hub d'un des étages est défectueux, cet étage ne peut plus communiquer avec les autres étages.

## 8.2.6 Supports de transmission

**Le câble coaxial**

Il y a différentes sortes de câble coaxial selon les applications.

Actuellement, ces câbles ne sont pratiquement plus utilisés dans les nouveaux câblages.

La vitesse maximale de transmission est de 10Mbps.

Les câbles coaxiaux les plus courants obéissent à la spécification RG 58.

Câble RG-58 A/U – Impédance = 50 ohms, câble coaxial fin Thinnet 10Base2

Ce câble Thinnet est également appelé « Cheapernet ». Dans un réseau en bus, la longueur de câble peut aller jusqu'à 185 m. Les réseaux ainsi câblés sont dénommés « réseaux 10Base2 ».

L'abréviation 10Base2 ainsi que 10Base5 correspond à des caractéristiques du réseau : vitesse de transmission de 10 Mbps dans la bande passante et distance maximale d'environ 200 m et 500 m par segment.

Il n'est pas possible de passer à une vitesse de transmission de 100 Mbps avec des câbles coaxiaux.

**Le câble à paire torsadée (twisted-pair cable)**

Ce sont des câbles avec des paires d'âmes isolées, entrelacées en torsade. L'entrelacement permet de supprimer jusqu'à un certain point les interférences d'origine extérieure ou dues aux paires adjacentes.

La vitesse de transmission peut aller jusqu'à 100Mbps et plus. La distance maximale entre l'ordinateur et le composant central (par ex. hub) est normalement de 100m.

**Paire torsadée non blindée UTP (unshielded twisted pair)**

Câble à paires torsadées sans blindage supplémentaire

- Sujet aux interférences !

**Paire torsadée blindée STP (shielded twisted pair)**

Chaque paire est protégée par un blindage.

- Beaucoup moins sujet aux interférences électriques.
- Les données sont transmises à une plus grande vitesse sur des distances plus longues.

PDM_UserMan_Netze_fr.fm

| Catégorie | Vitesse de transmission | Application |
|---|---|---|
| 1 | 1 Mbps | Transfert de voix analogique, systèmes d'alarme |
| 2 | 4 Mbps | Câblage IBM type 3 (voix), EIA-232 |
| 3 | 10 Mbps | 10Base-T, Token Ring 4 Mbit 100Base-T4, ISDN |
| 4 | 16 Mbps | Token Ring 16 Mbit |
| 5 et 5e | 100 Mbps | 100Base-Tx, ATM (155 Mbps), SONET, SDH |

*Tableau 8-1: Catégories de câbles réseaux*

Les autres catégories (6 et 7) de transmission en gigabits par fil de cuivre sont en cours de normalisation.



**REMARQUE**

Dans le cas d'utilisation d'un câble STP, avec une partie en UTP, l'avantage des câbles STP est annulé.

**Câble à fibre optique**

La transmission des signaux se fait sous la forme d'impulsions lumineuses envoyées dans une seule direction. C'est pourquoi il faut par câble au minimum deux brins (faisceaux). Les impulsions lumineuses sont générées par une diode laser ou une DEL et transmises dans le cœur.

- Grande vitesse de transmission (en gigabits) et distances maximales
- Bonne sécurité (contre interférences et espionnage)

Il faut distinguer deux sortes de fibres :

**Fibre monomode**

Dans le coeur de la fibre, la lumière se propage en ligne droite et les réflexions de lumière (dispersion) sont minimales.

- 100 GHz x km

**Fibre multimode**

Le coeur est plus gros que celui de la fibre monomode et la dispersion plus grande.

- 1 GHz x km

## 8.3 Adresse MAC

Chaque carte réseau a une adresse unique au monde attribuée par le constructeur.

L’architecture d’une adresse MAC (Media Access Control) est toujours la même.

| manufacturer | | | network card | | |
|---|---|---|---|---|---|
| 00 | 20 | AF | B7 | 32 | 29 |

*Illustration 8-6: Architecture de l’adresse MAC*

L’ adresse MAC est affichée en entrant *ipconfig /all* ou *GetMac* dans l’invite de commandes.

## 8.4 Protocoles d’accès

### 8.4.1 Le protocole CSMA/CD avec détection de collision

Le protocole d’accès d’Ethernet est connu sous le nom de CSMA/CD (Carrier Sense Multiple Access with Collision Detection).

Chaque station peut commencer à émettre quand elle le désire. Toutes les stations ont accès à tout moment et simultanément (multiple access) au support commun de transmission.

**Procédure d’émission :**

Quand une station émet des données, les signaux sont envoyés sur le câble dans les deux directions. Au cas où deux stations commencent à émettre en même temps, les signaux risquent d’être détruits (risque de collision). Les stations essaient de limiter ce risque en utilisant des procédés qui consistent à écouter le média de transmission avant d’émettre pour savoir si la porteuse est disponible. S’il y a déjà des signaux sur le câble, la station ne commence pas l’émission.

**Détection de collisions :**

Une collision peut toutefois se produire : en effet, deux stations ont détecté que le câble était « libre » et ont commencé à émettre en même temps. Il faut donc détecter cette collision pour y remédier.

Pour détecter une collision, toutes les stations continuent à écouter le média. La station qui détecte en premier la collision envoie un signal de brouillage (jam signal). Toutes les stations qui reçoivent ce signal arrêtent immédiatement d’émettre. Si la ligne est disponible à nouveau, les stations peuvent essayer d’émettre à nouveau. Soit un nouvel essai est tenté après un délai de durée aléatoire (non persistent CSMA), soit la station qui veut émettre écoute à nouveau le média et émet immédiatement

si le média est détecté comme étant libre (1-persistent CSMA).

- Plus il y a de stations connectées qui veulent émettre, plus les collisions sont fréquentes et plus le débit effectif de données est diminué.

PDM_UserMan_Netze_fr.fm

- L’instant de l’émission ne peut pas être calculé ni déterminé, mais survient de manière aléatoire.
- La procédure n’est pas adaptée aux applications où le temps est un élément critique.

## 8.4.2 Token Passing ou Passage du jeton

C’est le protocole d’accès des réseaux en anneau à jeton Token Ring (par exemple, réseau ArcNet).

Chaque station du réseau Token Ring est autorisé à émettre à un intervalle de temps défini et donc calculable. Ce protocole est de type déterministe.

**Procédure d’émission :**

Lorsqu’une station veut émettre, elle est obligée d’attendre d’intercepter le jeton circulant le long de l’anneau. Le jeton est une structure binaire (trame). Un bit dans la structure du jeton détermine l’état de ce dernier : état libre ou état occupé.

Si une station reçoit un jeton libre, elle change l'état du jeton en le rendant occupé, puis insère dans le jeton les informations concernant la source et le destinataire ainsi que les données qu’elle veut transmettre. Tout ce paquet de données devient une trame et est envoyé à la station suivante qui vérifie l’adresse du destinataire et renvoie le message sur l’anneau s’il ne lui est pas destiné.

**Réception**

Quand le message arrive au destinataire, il est copié. La station destinataire accuse réception du message en entrant une donnée dans la trame, puis renvoie le paquet modifié vers la station suivante.

Lorsque ce paquet arrive à la station émettrice, l’accusé de réception lui permet de savoir que le message a bien été transmis. La station émettrice modifie la trame du jeton, le rendant ainsi libre et l’envoyant à la prochaine station.

**Régulation des accès**

Hormis l’architecture de base des réseaux Token Ring, il y a aussi des règles pour assurer la sécurité du transfert de données. Ainsi, dans chaque anneau, une station aura un rôle de « contrôleur actif ». C’est généralement la première station qui est active dans un anneau. Si cette station n’est pas allumée, une procédure déterminée (claim token) permet de trouver une autre station pour ce rôle. Le contrôleur actif s’assure entre autres qu’il y a toujours un jeton ou une trame (jeton + données) en cours de validité.

- Le réseau peut avoir le nombre maximum de stations admises sans que le débit de données défini ne soit altéré.
- A l’aide du procédé NAUN (Nearest Active Upstream Neighbor), le protocole de passage de jeton peut détecter rapidement si une station du réseau pose des problèmes et la cerner. Cette station est retirée automatiquement de l’anneau.
- Ce protocole est adapté aux applications où le temps est un élément critique.

PDM_UserMan_Netze_fr.fm

## 8.5 Les systèmes d’exploitation de réseau

### 8.5.1 Netware de Novell

Le protocole réseau utilisé est le protocole propriétaire IPX/SPX. Depuis la version 4.1, le protocole TCP/IP est également supporté ; depuis la version 5.0, TCP/IP est le protocole standard.

### 8.5.2 Windows NT et Windows 2000 de Microsoft

**Le concept de domaine sous NT**

Les services d’annuaires sous Windows NT utilisent le concept de domaines. Un domaine est une unité de gestion administrative créée dans un serveur unique (PDC, Primary Domain Controller).

L’administration se fait par des comptes d’utilisateur qui sont stockés dans une base de données de comptes d’utilisateur sur le PDC.

La gestion de la base de données peut être faite par d’autres ordinateurs.

Active Directory : évolution du concept de domaines sous Windows 2000.

Windows 2000 intègre le concept de domaines dans les services d’annuaires ADS (Active Directory Services).

L’architecture et la gestion de réseaux plus importants sont beaucoup plus simples que sous Windows NT. NT 4.0 peuvent être intégrés sans grande difficulté dans Active Directory.

## 8.6 Le protocole TCP/IP

### 8.6.1 Comparaison des modèles de référence

TCP/IP est au départ une architecture réseau en 4 couches correspondant aux 7 couches du modèle de référence OSI :

| Couche | OSI | TCP/IP | Couche |
|---|---|---|---|
| 7 | Application | Application | 4 |
| 6 | Présentation | | |
| 5 | Session | | |
| 4 | Transport | Transport (TCP) | 3 |

| Couche | OSI | TCP/IP | Couche |
|---|---|---|---|
| 3 | Hôte-réseau | Internet (IP) | 2 |
| 2 | Liaison de données | Hôte-réseau | 1 |
| 1 | Physique | | |

*Tableau 8-2: Comparaison OSI <-> TCP/IP*

## 8.6.2 Principaux protocoles

A côté des deux protocoles TCP et IP qui sont à la base de la « famille TCP/IP », il existe un grand nombre de protocoles variés (>100)

| Protocole | Nom | Description |
|---|---|---|
| TCP | Transmission Control Protocol | Protocole de mise en place des liens logiques entre les applications. Il est implanté dans la couche transport et sert à sécuriser le transfert des données ; c'est un protocole orienté connexion. |
| UDP | User Datagramm Protocol | Egalement implanté dans la couche transport, le protocole UDP fonctionne en mode déconnecté. Il est donc moins fiable que le TCP, mais plus rapide. |
| IP | Internet Protocol | Protocole sans connexion et servant à diriger et transmettre les paquets via les adresses IP. |
| IPSec | IP Secure | Extension du protocole IP avec des mécanismes de sécurisation. |
| FTP | File Transfer Protocol | Protocole pour le transfert de fichiers entre ordinateurs ; il est rattaché à la couche application. |
| HTTP | Hypertext Transfer Protocol | Protocole régissant le transport de pages HTML (Hypertext Markup Language) et rattaché à la couche application. |
| Telnet | Telecommunication Network Protocol | Protocole d'émulation de terminal pour la communication avec un hôte. Telnet est rattaché à la couche application. |
| SMTP | Simple Mail Transfer Protocol | Protocole servant les transferts de courriers électroniques ; il est rattaché à la couche application. |
| POP | Post Office Protocol | Protocole permettant le téléchargement des e-mails stockés ; il est rattaché à la couche application. |

| Protocole | Nom | Description |
|---|---|---|
| ARP | Adress Resolution Protocol | Protocole permettant de traduire une adresse de type logique (adresse IP) en une adresse physique (adresse MAC). |

*Tableau 8-3: Présentation des principaux protocoles*

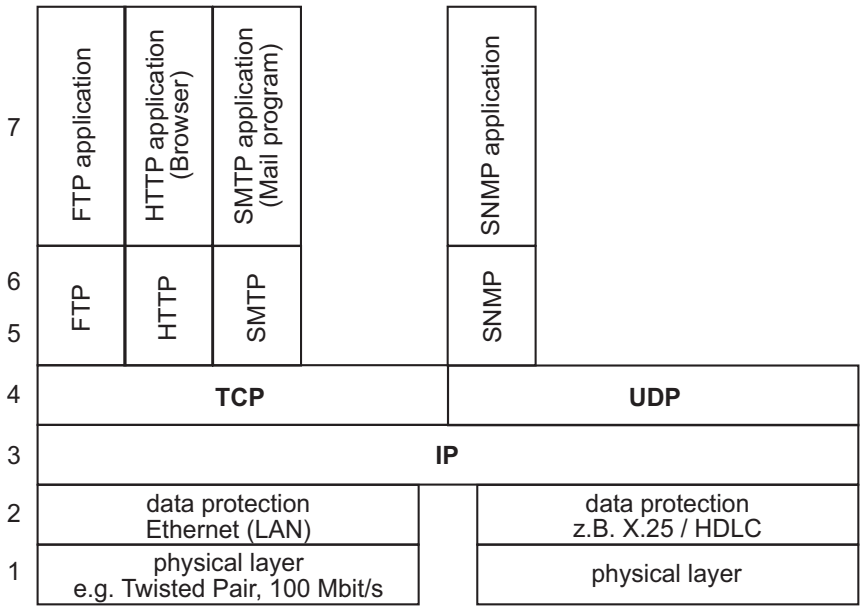



*Illustration 8-7: Architecture des protocoles et des services et classement en couches OSI*

**Numéro de port :**

Du fait que les différents services (FTP, Telnet, SMTP ou http) peuvent être appelés à la même adresse IP d ‘un ordinateur sous TCP, il faut une identification supplémentaire pour pouvoir distinguer quel service est demandé concrètement.

C’est le rôle du numéro de port.

| Numéro | Signification |
|---|---|
| 20 | Fichiers FTP |
| 21 | Commandes FTP |
| 23 | Telnet |
| 25 | SMTP |
| 80 | HTTP |
| 110 | POP3 |

*Tableau 8-4: Numéros de port*

PDM_UserMan_Netze_fr.fm

**Socket :**

L’adresse IP permet d’identifier clairement un ordinateur et le numéro de port de connaître le service appelé à l’ordinateur.

Ces deux informations regroupées sont appelées « socket ».

## 8.6.3 Adressage IP

Avec le protocole TCP/IP, chaque carte réseau doit obligatoirement se voir attribuer une adresse IP univoque à l’intérieur d’une interconnexion de réseaux. Des adresses IP logiques correspondent aux adresses physiques de cartes réseau (adresses MAC).

Pour les ordinateurs connectés à Internet, cela signifie que leurs adresses IP sont uniques dans le monde.

## 8.6.4 Masque de sous-réseau (subnet mask)

Le masque de sous-réseau doit obligatoirement être indiqué lors de l’installation du protocole réseau.

Lorsqu’un ordinateur du réseau reçoit un paquet de données, il vérifie par comparaison, à l’aide du masque de sous-réseau, que ce paquet est bien destiné à un ordinateur du segment de réseau. Si c’est le cas, il vérifie que les numéros de noeud sont bien les siens. Si tout concorde, il lit le contenu complet du paquet de données.

**Exemple**

| | **ID de réseau** | | | **ID d’hôte** |
|---|---|---|---|---|
| Adresse IP<br>- décimal<br>- binaire | <br>192<br>1100 0000 | <br>168<br>1010 1000 | <br>10<br>0000 1010 | <br>1<br>0000 0001 |
| Masque de sous-réseau<br>- décimal<br>- binaire | <br>255<br>1111 1111 | <br>255<br>1111 1111 | <br>255<br>1111 1111 | <br>0<br>0000 0000 |

*Tableau 8-5: Correspondances adresse IP et masque de sous-réseau*

## 8.6.5 Classes d’adresses IP

L’organisme central qui attribue les adresses IP s’appelle InterNIC (Internet Network Information Center).

En Allemagne, le NIC local est le DENIC (Deutsche Netzwerk-Informations-Center).

Ces organismes ne donnent toutefois pas d’adresses IP individuelles, mais des groupes d’adresses que les fournisseurs d’accès redistribuent à leurs clients. Les fournisseurs peuvent demander des nombres définis d’adresses IP. C’est ainsi que les adresses IP sont réparties en classes d’adresses.

Les classes A, B et C sont prévues pour l’adressage des ordinateurs.

| Classe | ID réseau | Subnet Mask | Nombre de réseaux | Nombre de noeuds de réseau |
|---|---|---|---|---|
| A | 0 à 126 | 255.0.0.0 | 126 | 16.777.216 |
| B | 128.0 à 191.255 | 255.255.0.0 | 16.384 | 65.536 |
| C | 192.0.0 à 223.255.255 | 255.255.255.0 | 2.097.152 | 256 |

*Tableau 8-6: Classes d’adresses réseau*

Les adresses qui commencent par 127 ne peuvent pas être attribuées car elles sont destinées à des fonctionnalités spécifiques. Ainsi l’adresse 127.0.0.1 est celle qui désigne la machine locale (localhost), adresse de boucle locale (loopback).

## 8.6.6 Réseaux privés IP

Pour la création de réseaux IP privés – cela concerne également de nombreux petits réseaux locaux (LAN) - , une plage d’adresses est retirée de chaque classe de réseau (voir ci-dessus).

Les adresses de ces trois plages ne sont jamais attribuées en tant qu’adresses Internet officielles et les routeurs ne les renvoient pas non plus vers l’Internet.

10.0.0.0 à 10.255.255.255 dans la plage de la classe A

172.16.0.0 à 172.31.255.255 dans la plage de la classe B

192.168.0.0 à 192.168.255.255 dans la plage de la classe C

Si un réseau local est adressable via Internet, il convient de vérifier que l’adresse IP est univoque dans tout l’Internet.

Dans ce cas il faut acquérir une adresse IP officielle.

## 8.6.7 Affectation d’adresse IP

Windows 2000 dispose de quatre procédures d’affectation d’adresses IP aux clients TCP/IP :

- DHCP
- APIPA
- Adressage IP statique
- Configuration IP alternative (multi réseau)

### DHCP

Le protocole DHCP (Dynamic Host Configuration Protocol) permet d’effectuer la configuration automatique d’adresses IP ainsi que d’autres configurations pour les clients d’un réseau comportant un ou plusieurs serveurs DHCP.

Dans le serveur DHCP, des plages IP (adresses IP à attribuer) sont créées et configurées avec les options correspondantes comme, par exemple, la passerelle par défaut ou le serveur DNS. Lors du démarrage, un client demande au serveur DHCP une configuration IP qu’il peut utiliser pendant un laps de temps défini (lease time ou bail). Lors de l’arrêt, le client ferme la session du serveur DHCP et ce dernier donne l’adresse IP à un autre client.

### APIPA

APIPA (Automatic Private IP Adressing) est utilisé pour les réseaux simples avec un seul sous-réseau. Si aucun serveur DHCP n’est disponible, l’ordinateur s’alloue automatiquement une adresse IP privée à l’aide d’APIPA. (plage : 169.254.0.1 à 169.254.255.254)

Si un serveur DHCP est à nouveau disponible et détermine une adresse, l’ordinateur modifie son adresse IP en conséquence.

### Adressage IP statique

L’adressage IP statique permet de configurer manuellement les adresses IP lorsque les fonctionnalités DHCP et APIPA ne sont pas disponibles ou n’existent pas.

Cette méthode exige du temps surtout si le réseau est important et n’est pas fiable.

### Configuration IP alternative (multi réseau)

La configuration IP alternative permet d’allouer plusieurs adresses à une carte réseau. Lorsqu’il faut connecter plusieurs réseaux (à des endroits différents par ex.) la carte réseau reçoit pour chaque sous-réseau une adresse IP particulière.

## 8.6.8 Résolution des noms

Le nom d'un ordinateur est converti ou « résolu » en une adresse IP qui lui est associée.

Windows 2000/XP propose quatre méthodes de résolution de nom en adresse IP.

Il s'agit de DNS, de WINS, de la résolution de noms avec les fichiers HOSTS ou LMHOSTS et de la résolution de noms par broadcast. Windows 2000/XP utilise en général une combinaison de ces méthodes.

**DNS**

Des requêtes sont envoyées aux serveurs DNS. La méthode est utilisée pour les applications et les services qui ont besoin d'une résolution de noms hôte - IP. Cette méthode est appelée quand l'ordinateur veut accéder à l'Internet.

DNS est la méthode par défaut de résolutions de noms pour les clients de réseau sous Windows XP.

Si le service de résolution de Windows XP obtient une réponse positive ou négative à une requête, il enregistre cette réponse dans un cache et crée ainsi une ressource DNS. Le service de résolution vérifie toujours le cache avant la requête des serveurs DNS. Si une ressource DNS se trouve dans le cache, le service de résolution utilise cet enregistrement et n'effectue plus de requête auprès du serveur. Cette procédure accélère les requêtes et le trafic sur le réseau concernant les requêtes DNS est réduit.

**Afficher le cache de résolution DNS :**

```
ipconfig /displaydns
```

**Vider manuellement le cache DNS :**

```
Ipconfig /flushdns
```

**Résolution de noms NetBIOS**

Des requêtes sont envoyées au serveurs WINS. Cette méthode a été adoptée, pour des raisons de compatibilité, pour les applications et les services qui ont besoin de la résolution de noms NetBIOS - IP, comme par exemple la fonction service de recherche de Windows NT 4.0 et Windows 98/95.

## 8.6.9 Résolution de noms NetBIOS et IP Host

Elle est mise en oeuvre par des fichiers HOSTS ou LMHOSTS.

Ces fichiers mettent à disposition une résolution hôte/IP et NetBIOS/IP à l'aide de fichiers gérés localement.

**Résolution de noms NetBIOS (broadcast)**

Elle est mise en oeuvre à l'aide du broadcast (noeud B). La résolution de noms est exécutée par le broadcast à l'intérieur du sous-réseau local.

## 8.7 Interconnexion de réseaux

### 8.7.1 Les équipements d'interconnexion

Pour agrandir un réseau il est nécessaire d'ajouter des équipements qui vont permettre de gérer les nouvelles contraintes.

Certains équipements servent uniquement à relier plusieurs segments ensemble (sous-domaines d'un réseau particulier). D'autres peuvent par contre connecter plusieurs réseaux autonomes (sous-réseau).

**Répéteur**

Un répéteur (amplificateur de signal) est un composant actif du réseau avec une fonctionnalité de régénération.

Un répéteur reçoit des signaux, les amplifie et les renvoie.

Son rôle, dans les réseaux locaux, est d'augmenter la portée maximale des signaux et par là, la longueur de câble maximale. Un répéteur permet ainsi de relier des segments de réseau d'un type de LAN identique sur la base de la transmission de données.

Un répéteur agit sur la couche 1 du modèle OSI, c'est-à-dire qu'il se borne à régénérer les flux de bits reçus qu'il réémet sans se soucier du contenu des trames. Cela implique que les segments concernés aient le même protocole d'accès au média de transmission. Un répéteur ne peut pas relier par exemple des segments sous Ethernet et des segments sous Token Ring.

Appareil de niveau 1, le répéteur est totalement « transparent » : ni les autres appareils raccordés ni les programmes ne « voient » le répéteur.

Les répéteurs n'ont plus de rôle déterminant dans les réseaux de dernière génération.

**HUB**

Les hubs agissent, comme les répéteurs, sur la couche 1 du modèle OSI et sont totalement transparents pour les autres systèmes.

La connexion de plusieurs hubs permet d'augmenter sans trop de difficulté le nombre de stations raccordées.

Les hubs sont connectés entre eux avec un câble croisé. Si un câble droit est utilisé, il faut utiliser une entrée spéciale sur le hub, l'entrée « Uplink ».

Les terminaux sont chacun connectés au hub avec un câble droit. Cette interface est appelée « port ».

Du fait que chaque appareil connecté dispose d'une liaison dédiée directe vers le hub via un câble qui lui est propre, les hubs offrent à ces appareils la totalité de la bande passante du réseau local au moins jusqu'au port du hub. Toutefois l'ensemble du segment et les appareils raccordés sur ce segment doivent partager le média de transmission : chaque noeud reçoit chaque message, y compris les messages destinés à d'autres stations.

**Pont**

Un pont (ou bridge) agit déjà sur la couche 2 du modèle OSI (niveau adresse MAC) et il est transparent pour les couches de protocole supérieures. Cela signifie que tous les protocoles travaillant sur la couche 2 passent par le pont sans modification. A la différence du répéteur, un pont peut changer les différentes vitesses de transmission ainsi que les divers protocoles d'accès ; il sera par exemple implanté entre un protocole CSMA/CD et un protocole Token Passing. Un pont relie des segments de réseau ayant des technologies différentes Ethernet et Token-Ring par exemple) contrairement au répéteur. Si le pont connaît le destinataire, il n'envoie les données que dans le réseau concerné.

Un pont ne peut pas limiter à un segment des diffusions générales sous forme de broadcast ; en d'autres termes, des diffusions générales d'un ordinateur à tous les autres sont transmises aux autres segments.

Un pont enregistre temporairement et prépare les paquets de données avant de les renvoyer à l'aide de l'adresse MAC. Il utilise un répertoire d'adresses (tables du pont), où sont entrées les adresses MAC des stations interconnectées.

**Commutateur**

Un commutateur ou switch est en premier lieu le prolongement d'un pont. Comme ce dernier, le commutateur travaille sur la couche 2 OSI et utilise les adresses MAC pour diriger les données.

Un pont est capable d'améliorer la bande passante sur l'ensemble du réseau car il redirige les données uniquement s'il s'agit d'une transmission allant d'un segment à un autre.

Un commutateur par contre peut améliorer la bande passante à l'intérieur d'un segment.

A la différence du pont, des connexions directes et dédiées sont activées entre les ports. Les paquets de données ne sont envoyés qu'au port de la station destinataire. La charge du réseau est ainsi réduite.

Le commutateur travaillant avec le niveau des adresses MAC, il ne peut pas coupler deux segments de réseau entre eux.

Certains constructeurs proposent aussi des commutateurs hautes performances qui travaillent non seulement sur la couche 2 mais aussi sur la couche 3, voire plus.

**Commutateur couche 3 – Commutateur couche 4**

Un commutateur de couche 2 ne dispose pour l'envoi des paquets que de l'adresse MAC. Un commutateur de couche 3 va utiliser en plus l'adresse IP destinataire et un commutateur de couche 4 dirigera les données avec un numéro de port.

**Routeur**

Un routeur est un appareil qui interconnecte des réseaux séparés ou divise de grands réseaux en sous-réseaux.

La forme basique d'un routeur est un PC équipé de plusieurs cartes réseau ayant chacune un contact avec les différents réseaux.

Les routeurs agissent sur la couche 3 du modèle OSI. Ils peuvent ainsi relier des réseaux de topologies différentes des couches inférieures 1 et 2.

Il faut pour cela que tous les réseaux concernés utilisent le même type d'adressage pour leurs paquets de données.

Le routeur décapsule les paquets de données entrants jusqu'à la couche 3 afin de trouver dans l'en-tête (header) l'adresse destinataire correspondante. Cette procédure prend du temps et les routeurs sont généralement plus lents que les commutateurs ou les ponts.

Le classement des routeurs dans la couche 3 OSI indique en outre qu'un routeur utilise comme destination non pas l'adresse MAC, mais l'adresse du niveau de protocole (adresse IP par exemple).

**Tables de routage**

La fonction principale d'un routeur est de trouver un chemin (routing) pour les paquets de données (datagrammes) allant de leur source à leur destination.

Les routeurs se servent pour cela de tables de routage où sont entrées les adresses connues des réseaux destinataires.

Lorsqu'un paquet arrive et que le routeur a lu l'adresse de destination voulue, ce dernier consulte la table de routage pour vérifier si cette adresse est entrée.

Si c'est le cas, le routeur peut envoyer le paquet. Dans le cas contraire, il envoie le paquet à un autre routeur (routeur par défaut ou passerelle par défaut/default gateway) qui cherchera à sa manière à rediriger le paquet. Cette méthode est aussi appelée routage par sauts successifs (next hop routing).

La passerelle à entrer dans le panneau de configuration de Windows est en fait un routeur.

**Routage sous Windows 2000**

Sur chaque ordinateur installé sous Windows 2000/XP, la table de routage statique des segments suivants est automatiquement créée. Pour cela, le système analyse pour chaque carte réseau installée et configurée les informations effectives de configuration du protocole IP. Il faut que le service de routage et d'accès à distance soit démarré (voir PacDrive P et Ethernet).

Pour tester les propriétés de routage d'un ordinateur sous Windows2000/XP, divers outils sont à votre disposition et ils sont seront évoqués en fin de chapitre.

**Passerelle**

Une passerelle ou gateway relie des réseaux ayant des adresses et des protocoles totalement différents.

Une passerelle permet de faire communiquer des réseaux a priori incompatibles. Pour cela, elle va intervenir sur les 7 couches OSI et dans le cas le plus extrême, fragmenter un message entrant jusqu'à la couche 7 pour le ré-assembler et retourner à la couche 1 afin qu'il soit compatible pour l'autre réseau.

Une passerelle convertit réellement un protocole en un autre protocole.

## 8.7.2 Outils de gestion des problèmes TCP/IP

**Ping (ping adresse IP)**

Ping est un utilitaire permettant de vérifier la connectivité IP. La commande Ping permet d'envoyer une requête d'Echo ICMP (Internet Control Message Protocol) à un hôte destinataire. Il convient d'utiliser Ping quand il faut vérifier qu'un hôte peut envoyer des paquets IP à un hôte destinataire.

```
Ping adresse IP de l'hôte distant
Ping nom d'hôte de l'hôte distant
```

Ping utilise la résolution de noms pour traduire un nom d'ordinateur en adresse IP. Si le signal Ping peut être envoyé avec succès à une adresse IP, mais que l'envoi du signal à un nom échoue, la résolution de nom d'hôte en est la cause – et non la connectivité du réseau.

En cas d'échec total avec Ping, vérifiez que les conditions suivantes sont bien réunies :

- L'adresse IP de l'ordinateur local est valide et s'affiche correctement soit sur l'onglet GENERAL de la boîte de dialogue « Internet protocol properties (Propriétés Protocole Internet) (TCP/IP) », soit en appelant IPCONFIG.
- Une passerelle par défaut a été configurée et la connexion entre celle-ci et l'hôte est disponible. Depuis Windows 2000, il est certes possible de configurer plusieurs passerelles par défaut. Or, les

passerelles sont utilisées après la première passerelle uniquement lorsque le paquet IP constate que la passerelle d'origine ne fonctionne pas. Dans ce cas, il sera plus simple d'effacer toutes les autres passerelles.



**REMARQUE**

Lorsqu'une connexion au système distant, à qui est envoyé le signal Ping, présente un retard important - connexion par satellite par ex. -, la réponse peut durer plus longtemps. Le paramètre – w (wait) permet d'augmenter la durée d'attente, le délai d'attente par défaut étant de 4 secondes.

Messages d'erreurs renvoyés fréquemment par Ping :

| Message d'erreur | Signification et action |
|---|---|
| Time to live exceeded during transmission (La durée de vie est dépassée lors de la transmission) | Le nombre de sauts (hop count) dépasse la durée de vie (TTL = Time to Live).<br>Prolongez la durée de vie avec le paramètre *ping -i*. |
| Target host cannot be reached (Impossible de joindre l'hôte destinataire | Il n'y pas pas de route locale ou distante pour l'hôte destinataire dans l'hôte émetteur ou dans le routeur.<br>Mettre à jour la table de routage dans l'hôte local ou dans le routeur (*route add*). |
| Request time exceeded (Délai d'attente de la demande dépassé) | Dans l'intervalle défini (délai standard : 4 secondes), aucun message de réponse à écho n'est reçu.<br>Augmentez le délai avec le paramètre *ping -w.* |
| Ping request could not find host (Ping ne trouve pas l'hôte) | Le nom de l'hôte destinataire ne peut pas être résolu.<br>Vérifiez le nom et la disponibilité des serveurs DNS ou WINS. |

*Tableau 8-7: Signification et aide lors des messages d'erreur de la commande réseaux Ping*

**Vider la mémoire-cache ARP (Address Resolution Protocol)**

Lorsqu'un signal Ping ne peut pas être envoyé avec succès à l'adresse de boucle locale 127.0.0.1 (loop back) ni à l'adresse IP propre, ni à aucune autre adresse IP, il faut éventuellement vider la mémoire cache ARP.

L'utilitaire ARP est alors utilisé. Les commandes arp –a et arp -g affichent le contenu du cache. Les entrées sont supprimées avec la commande *arp –d IP- Adresse*.

La mémoire ARP entière peut être vidée avec la commande *netsh interface ip delete arpcache*.

### Route print

La commande « route print » permet d'afficher la table de routage.

Pour que deux hôtes IP puissent échanger des datagrammes, ils doivent disposer d'un itinéraire (route) les reliant entre eux ou bien ils doivent utiliser une passerelle par défaut connaissant une route.

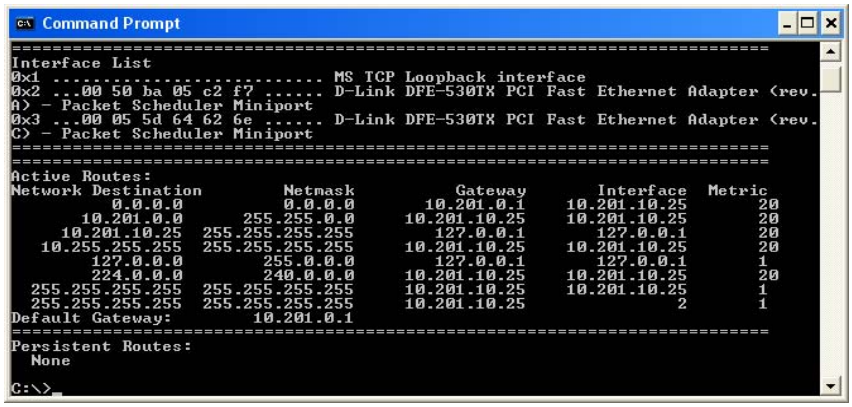



*Illustration 8-8: La commande « route print » dans l'invite de commande MS-Dos*

La table de routage est celle d'un ordinateur avec l'adresse IP 10.201.17.124, le masque de sous-réseau 255.255.0.0 et la passerelle par défaut 10.201.0.1. Elle comporte les entrées suivantes :

**Destination réseau**

L'adresse de réseau de la table de routage est l'adresse de destination. La colonne de la destination réseau peut comporter trois types d'entrée différents disposés par ordre de précision.

1) Adresse hôte (itinéraire pour une adresse IP cible spécifique)

2) Adresse sous-réseau (itinéraire pour un sous-réseau)

3) Itinéraire par défaut 0.0.0.0 (utilisé lorsqu'il n'y a pas d'autre correspondance)

4) Si aucun itinéraire ne correspond, les paquets sont détruits.

Voici le détail des entrées de la table de routage :

1. Itinéraire par défaut utilisé pour le transfert de paquets à toutes les destinations excepté le sous-réseau local.

2) Itinéraire sous-réseau pour le sous-réseau local.

3) Itinéraire hôte pour l'hôte local (itinéraire pour les paquets envoyés à cet ordinateur hôte)

4) Itinéraire hôte pour une adresse broadcast IP spécifique, adresse broadcast pour tous les sous-réseaux.

5) Itinéraire local « loopback » utilisé par un hôte pour envoyer des paquets à sa propre adresse.

6) Itinéraire multicast IP que l'ordinateur utilise pour envoyer des paquets à un groupe multicast IP.

7) Itinéraire hôte pour l'adresse broadcast réservée.

**Masque de réseau**

Le masque de réseau indique quelle partie de l'adresse de destination doit correspondre à l'adresse du réseau afin que la route correspondante soit interprétée comme associée à l'adresse de destination du paquet transmis. Si le masque est en mode binaire, la valeur 1 définit un bit associé et la valeur 0 définit un bit non associé.

Dans le cas d'un masque comportant par exemple exclusivement des 1 (255.255. 255. 255), l'adresse de destination du paquet à transmettre doit correspondre exactement à l'adresse complète de l'hôte afin que cet itinéraire soit considéré comme associé. Dans l'exemple du masque 255.255.0.0 les deux premiers octets doivent obligatoirement concorder exactement. La concordance du dernier octet n'est pas obligatoire.

**Adresse passerelle**

Cette fonction spécifie dans la table de routage l'adresse IP de transmission. Il s'agira soit de l'adresse IP même de l'hôte ou celle d'un réseau dans le sous-réseau local. Si l'adresse passerelle de la route est l'adresse IP de l'hôte, l'adresse IP sera fixée dans le datagramme pour la transmission vers l'adresse IP de destination. C'est l'adresse IP de l'hôte qui sera contactée par priorité. Que l'hôte se trouve dans le sous-réseau local ou dans un sous-réseau distant ne joue aucun rôle. Si l'adresse passerelle de l'itinéraire n'est pas l'adresse IP de l'hôte, c'est normalement l'adresse d'un routeur dans le sous-réseau local qui sera utilisée comme adresse IP de passerelle pour la transmission.

**Route add**

Lorsqu'il manque à deux hôtes connectables un itinéraire approprié, la commande « Route Add » permet d'ajouter un itinéraire.

Exemple :

```
ROUTE ADD 192.168.3.0 MASK 255.255.255.0 192.168.2.2
```

Avec l'option MASK, il est possible de « router » tous les réseaux : pour faire une requête à un hôte dans le réseau 192.168.3 par exemple, il faut passer par le routeur 192.168.2.2.

PDM_UserMan_Netze_fr.fm

### Tracert

La commande Tracert permet de suivre un itinéraire allant à une destination IP déterminée en donnant jusqu'à 30 sauts ou hops (routeurs et/ou passerelles).

La rapidité de la transmission de données est également tracée sur les tronçons concernés.

Exemple : tracert 10.205.0.1 indique le l'itinéraire du paquet menant vers la destination 0.205.0.1.

### Pathping

La commande Pathping envoie des paquets de données à chaque routeur se trouvant sur la route de la dernière adresse IP de destination réseau et identifie l'itinéraire d'un routeur à l'autre.

Pathping, en indiquant la quantité perdue de paquets sur des segments ou des connexions de routeur, permet de savoir exactement quels routeurs ou quelles connexions sont surchargées et provoquent des problèmes dans le réseau.

### Ordre d'accès aux connexions réseau

Lorsque, dans un ordinateur sous Windows XP Professional, plusieurs cartes réseaux et plusieurs protocoles réseaux sont implémentés, il est possible de définir l' ordre d'accès des connexions à chaque carte réseau et à chaque protocole pour les services utilisés par les protocoles respectifs.

Il définit quel protocole utilise un service pour la connexion à un autre service ou client, ou bien quelle carte réseau est utilisée pour la connexion.

Afin de diminuer la durée de la recherche des clients et des services nécessaires, il convient de placer en tête la carte réseau et le protocole les plus fréquemment utilisés.

De nombreux services peuvent créer des liens avec tous les protocoles.

L'accès au réseau est toutefois géré par le service client pour les réseaux Microsoft. L'ordre d'accès aux connexions est affiché, pour un adaptateur réseau sélectionné, en ouvrant la boîte de dialogue Connexion réseau avec un clic droit sur le Network environment (voisinage réseau), puis (Network environment | Advanced | Advanced settings (Propriétés | Avancé | Paramètres avancés) : allez sur l'onglet Network cards and connections (Cartes et Liaisons).

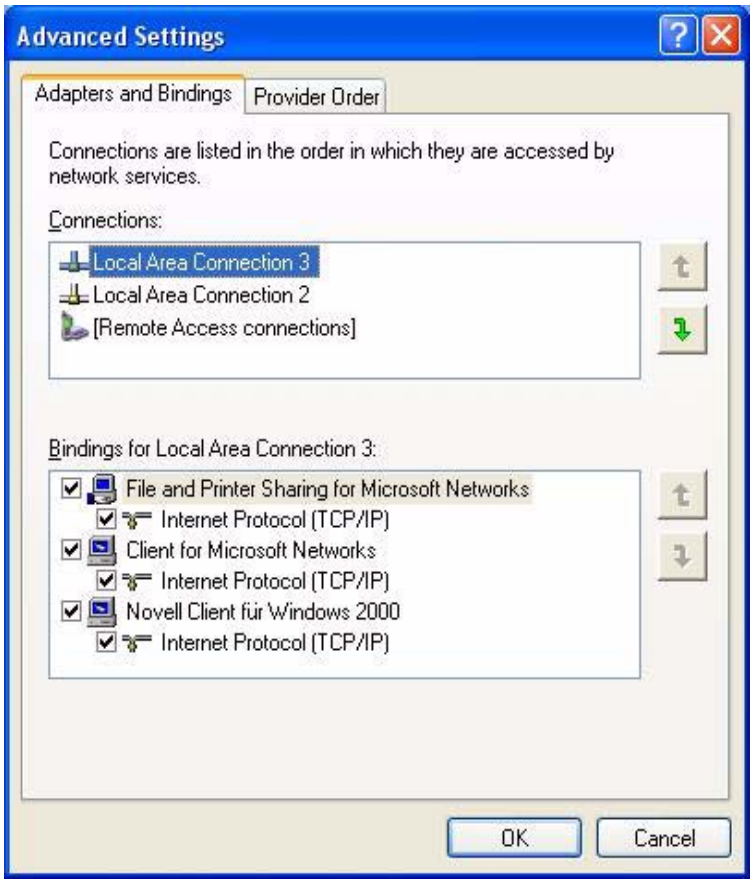



*Illustration 8-9: Configuration de l'ordre d'accès aux connexions réseau et aux protocoles*

### Observateur d'évènements

Succession des fenêtres :

Workplace | Settings | System programs | Events | Safety (Poste de travail | Gérer | Programmes système | Observateur d'évènements | Sécurité)

*Illustration 8-10: Boîte de dialogue : Event Properties (Propriétés de l'évènement)*

L'observateur d'évènements permet de surveiller les évènements se produisant dans le système. Le service des protocoles d'évènements est lancé par défaut lors du démarrage de Windows XP.

Une entrée type de protocole est composée de l'en-tête et de la description.

PDM_UserMan_Netze_fr.fm

# 9 Caractéristiques du système

| Grandeur caractéristique | Valeur |
|---|---|
| **Tâches IEC** | 64 au maximum |
| **Code de programme IEC (Code Size)** | 2 Mo |
| **Plage de données IEC (Data Size)** | 2 Mo<br>(dans des versions < V00.07.00 1 Mo) |
| Plage de variables IEC<br><br>Mémoire interne<br>Entrées<br>Sorties | Data Size - 3 x 64 ko<br><br>64 ko<br>64 ko<br>64 ko |
| **Variable de rétention IEC (IEC Retain Variable)** | |
| NvRam MAx-4<br>- Taille<br>- Rémanence des données | <br>1 ko<br>7 jours au maximum |
| Flash Disk MAx-4 avec objet USV<br>- Taille<br>- Rémanence des données<br>- Cycles d'écriture | <br>64 ko<br>> 100 ans<br>100 000 au maximum |
| NvRam sur module PN-4<br>- Taille<br>- Rémanence des données<br>- Sauvegarde par batterie | <br>101 ko<br>7 jours au maximum (sans batterie)<br>interne ou externe possible |
| **Tailles maximum de variables** | |
| Tableau | 128 ko (32 ko < V00.12.00) |
| Struct | 128 ko (32 ko < V00.12.00) |
| Liste des variables globales | 128 ko (32 ko < V00.12.00) |
| Taille de toutes les variables d'une UOP du type FB (Bloc fonctionnel) et Programme (UOP appelées inclues) | 128 ko (32 ko < V00.12.00) |
| **Profondeur de pile pour des fonctions** | 10 ko |
| **Niveaux d'imbrication d'instructions IF** | 15 |
| **Temps du cycle SERCOS** | |
| Précision | < 0,01 % |
| Résolution | 1, 2, 4 ms (réglable via le paramètre *CycleTime*) |

PDM_UserMan_Kennwerte_fr.fm

| Grandeur caractéristique | Valeur |
|---|---|
| **Horloge système** | |
| Précision | < 0,1 % |
| Résolution | 250 µs |
| Horloge temps réel matériel sur module PN-4<br>- Précision<br>- Rémanence des données (sans batterie)<br>- Sauvegarde par batterie | <br>±1 minute dans 12 jours<br>7 jours au maximum<br>interne ou externe possible |

*Tableau 9-1: Grandeurs caractéristiques du système PacDrive™*

**REMARQUE**

Des variables bit, déclarés avec le mot clé AT, ne peuvent être placées que sur les 4096 octets premiers. Autrement, une erreur de syntaxe se produit lors de la déclaration.
Le comportement d'initialisation peut être influencée à l'aide du paramètre objet MAx-4.General.RetainInit.
Lors de la compilation réussie du projet, les grandeurs *Code Size* et *Data Size* sont affichés.

# 10 Identification de version et compatibilité

## 10.1 Identification générale de version

VXX.YY.ZZ

XX - modifications non compatibles

YY - modification à compatibilité ascendante

ZZ - modifications compatibles

## 10.2 Compatibilité EPAS-4 - MAx-4

- EPAS-4 XX doit être égal à MAx-4 XX
- EPAS-4 YY doit être supérieur ou égal à MAx-4 YY
- EPAS-4 ZZ n'a pas de signification

## 10.3 Compatibilité MAx-4 - MC-4

- MAx-4 XX doit être égal à MC-4 XX
- MAx-4 YY doit être supérieur ou égal à MC-4 YY
- MAx-4 ZZ n'a pas de signification

## 10.4 Compatibilité MAx-4 - Bibliothèques IEC (IEC-Libraries)

- MAx-4 XX doit être égal à Lib XX
- MAx-4 YY doit être supérieur ou égal à Lib YY
- MAx-4 ZZ n'a pas de signification

PDM_UserMan_Versionsk_fr.fm

# 11 Diagrammes des temps – drive «MotorController»

## 11.1 Mise en marche de la validation

Au moment de la mise en marche de la validation, le drive « MotorController » adopte le comportement temporel suivant :

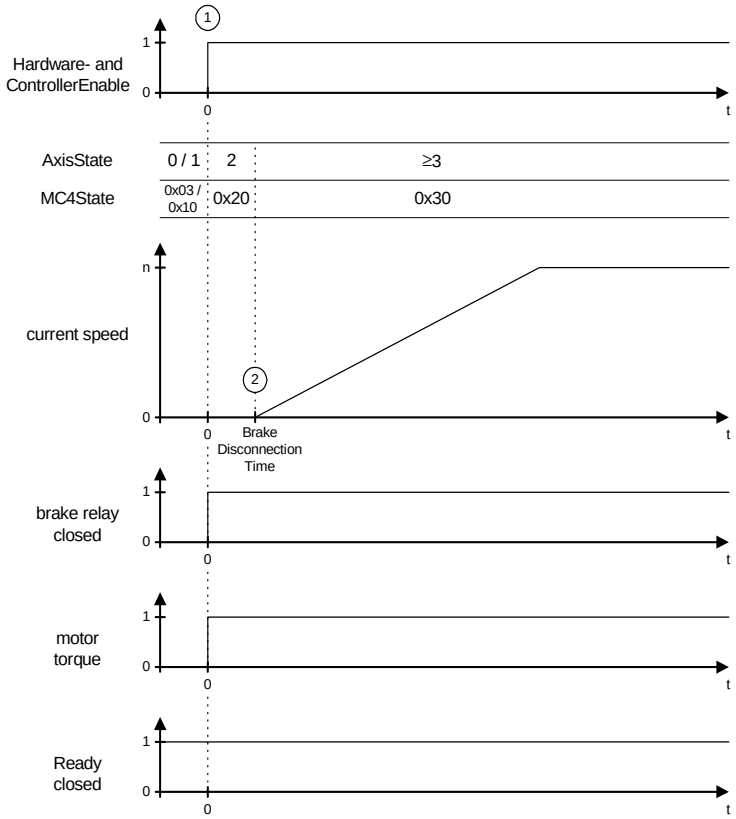



*Illustration 11-1: Diagramme des temps après la mise en marche de la validation*

Avant la mise en marche de la validation, *AxisState* et/ou *MC4State* dépendent du signal de validation attendu. Sans validation matérielle, *AxisState* a la valeur 0 et *MC4State* 0x03. Si l'entraînement attend la validation par *ControllerEnable*, *AxisState* est de 1 et *MC4State* de 0x10.
Pendant la validation matérielle et la transmission du signal *ControllerEnable* (temps 1), le relais de frein est fermé (frein relevé) et le couple moteur est enclenché, à condition toutefois qu'il n'y ait aucun message d'erreur, que le circuit intermédiaire soit chargé et que toutes les phases soient connectées. Après l'expiration du délai *BrakeDisconnectionTime* (temps 2 ; valeur par défaut : 100 ms), des instructions de mouvement peuvent être transmises à l'entraînement.

PDM_UserMan_ZeitDiag_fr.fm

## 11.2 Arrêt de la validation

Au moment de l'arrêt de la validation, il peut y avoir trois séquences d'événements.

### 11.2.1 Décélération avant expiration du temps de décélération maximal

Au moment de l'arrêt de la validation, le drive « MotorController » adopte le comportement temporel suivant si la décélération intervient avant la fin du temps de décélération maximal :

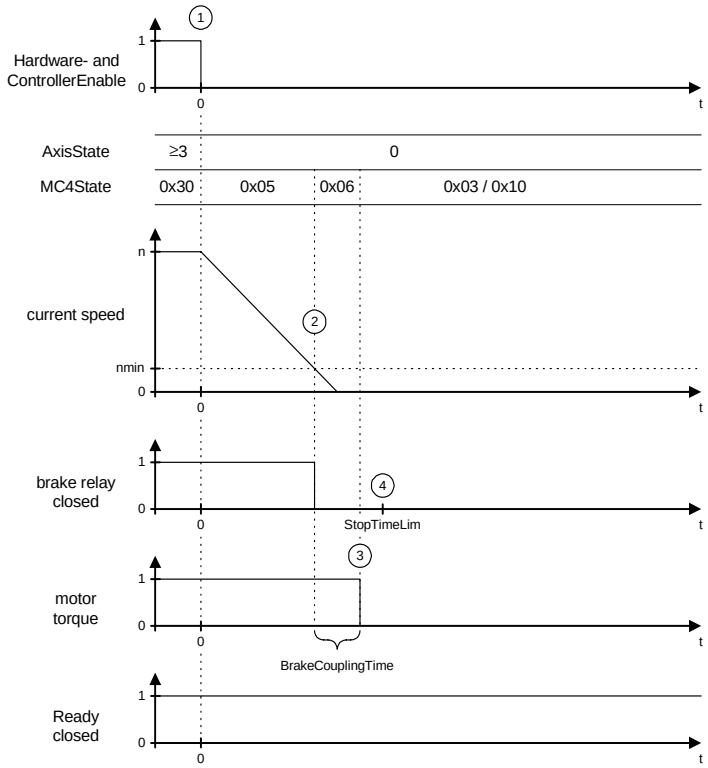



*Illustration 11-2: Diagramme des temps pour la décélération avant expiration du temps de décélération maximal*

Au moment de l'arrêt de la validation (temps 1), l'entraînement décélère avec un niveau de courant maximal. L'entraînement s'arrête avant expiration du temps de décélération maximal *StopTimeLim* (temps 4). Dès que la fréquence de rotation effective passe en-dessous du seuil inférieur (temps 2 : fréquence de rotation effective < nmin (10 $min^{-1}$), le relais de frein s'ouvre. Après l'expiration du temps d'enclenchement du frein BrakeCouplingTime (temps 3), le couple moteur est mis à zéro. Le contact Ready reste fermé.
La décélération s'est donc déroulée sans erreur, si la *CoastOption*, qui arrête la validation, est restée inactive (*bEnableCoast* = FALSE).

PDM_UserMan_ZeitDiag_fr.fm

### 11.2.2 Arrêt réglé de l'entraînement après dépassement du temps de décélération maximal

Au moment de l'arrêt de la validation, le drive « MotorController » passe au comportement temporel suivant si la décélération ne s'achève pas avant l'expiration du temps de décélération maximal, mais si l'entraînement peut encore être arrêté de façon réglée :

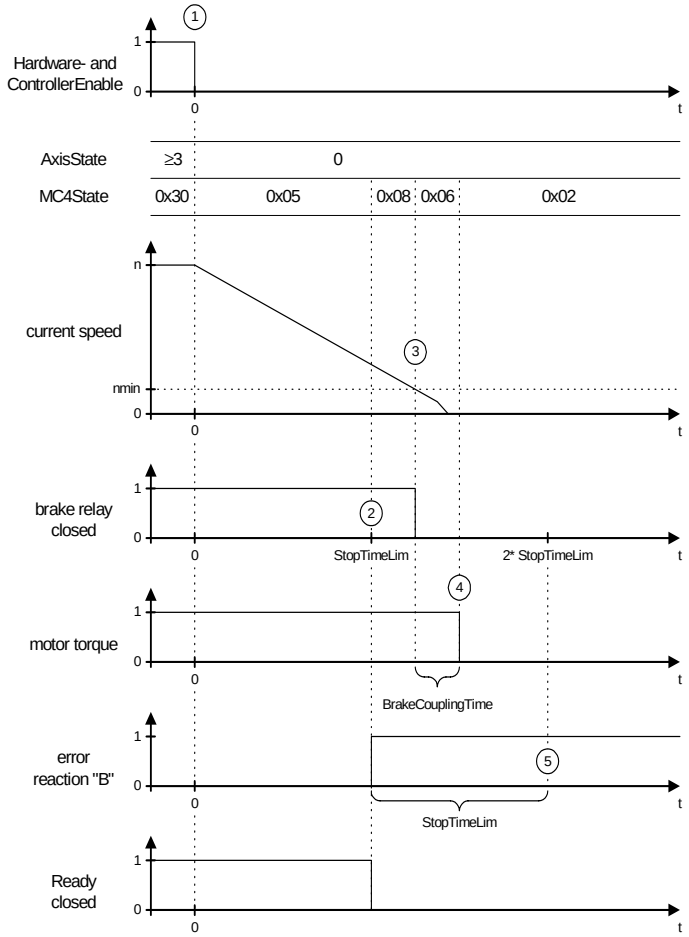



*Illustration 11-3: Diagramme des temps décélération (dépassement du temps de décélération maximal, arrêt)*

Au moment de l'arrêt de la validation (temps 1), l'entraînement décélère avec un niveau de courant maximal. Mais l'entraînement ne s'arrête pas avant l'expiration du temps de décélération maximal *StopTimeLim* (temps 2). Ce dépassement donne lieu au message d'erreur « Stop time limit exceeded » suivi de la réaction « B ». La réaction d'erreur « B » consiste dans l'ouverture du contact Ready et l'arrêt réglé de l'entraînement à un niveau de courant maximal. Si l'entraînement s'arrête avant l'expiration du temps de décélération maximal spécifié pour la réaction d'erreur « B » (temps 5) (temps 3 : fréquence de rotation effective < nmin), le relais de frein s'ouvre (le

frein s'enclenche) et le couple moteur est mis à zéro après l'expiration du délai d'enclenchement du frein *BrakeCouplingTime* (temps 4). La décélération reste donc réglée si l'entraînement s'arrête dans un délai correspondant au double du temps de décélération augmenté du délai d'enclenchement du frein et si la *CoastOption* n'est active ni pour l'arrêt de la validation ni pour la réaction d'erreur « B » (*bEnableCoast* = FALSE; *bErrorCoast* = FALSE).

### 11.2.3 Arrêt non réglé de l'entraînement après dépassement du temps de décélération maximal

Au moment de l'arrêt de la validation, le drive « MotorController » passe au comportement temporel suivant si l'arrêt de l'entraînement n'intervient pas avant l'expiration du temps de décélération maximal et si l'entraînement ne peut être arrêté de façon réglée :

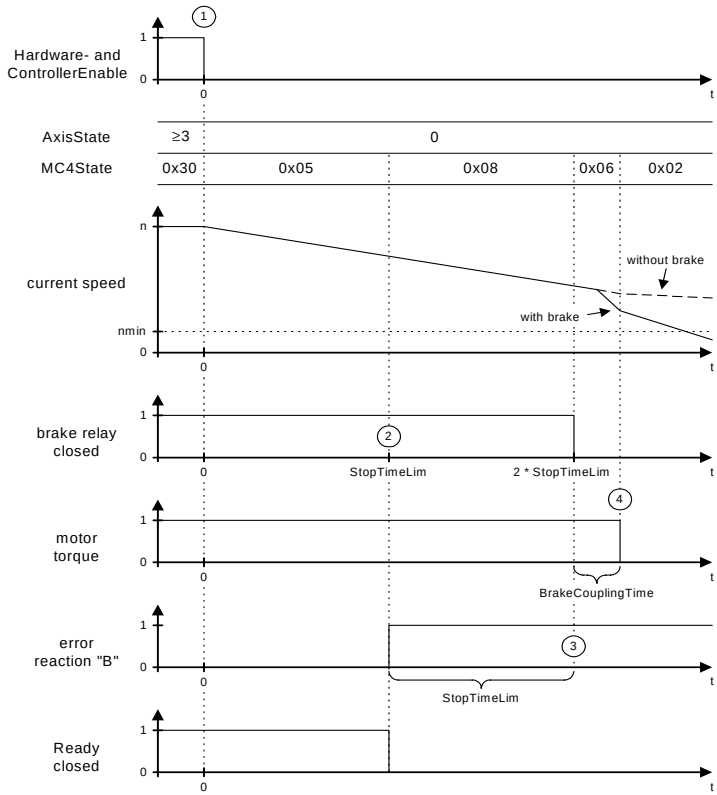



*Illustration 11-4: Diagramme des temps décélération (dépassement du temps de décélération maximal, arrêt par inertie)*

Au moment de l'arrêt de la validation (temps 1), l'entraînement décélère avec un niveau de courant maximal. Mais l'entraînement ne s'arrête pas avant l'expiration du temps de décélération maximal *StopTimeLim* (temps 2). Ce dépassement donne lieu au message d'erreur « Stop time limit exceeded » suivi de la réaction « B ». La réaction d'erreur « B » consiste dans l'ouverture du contact Ready

et l’arrêt réglé de l’entraînement à un niveau de courant maximal. Si l’entraînement ne s’arrête pas avant l’expiration du temps de décélération maximal spécifié pour la réaction d’erreur « B » (temps 3), le relais de frein est ouvert (enclenchement du frein) et le couple moteur, mis à zéro après l’expiration du délai d’enclenchement de frein *BrakeCouplingTime* (temps 4). L’entraînement est donc décéléré de façon réglée pendant un délai correspondant au double du temps de décélération maximal augmenté du délai d’enclenchement du frein, avant de s’arrêter de façon non réglée (par inertie). Mais ceci ne vaut qu’à condition que la *CoastOption* ne soit active ni pour l’arrêt de la validation ni pour la réaction d’erreur « B » (*bEnableCoast* = FALSE; *bErrorCoast* = FALSE). L’arrêt non réglé de l’entraînement ne donne pas lieu à la génération d’un message d’erreur.
Le comportement d’un entraînement arrêté de façon non réglée dépend, entre autres, de la présence d’un frein d’arrêt sur le moteur.

## 11.3 La réaction d’erreur « B »

Une erreur qui déclenche la réaction « B » peut donner lieu à deux séquences d’événements.

### 11.3.1 Décélération avant expiration du temps de décélération maximal

Une erreur déclenchant la réaction « B » donne lieu au comportement temporel suivant si la décélération intervient avant l’expiration du temps de décélération maximal :

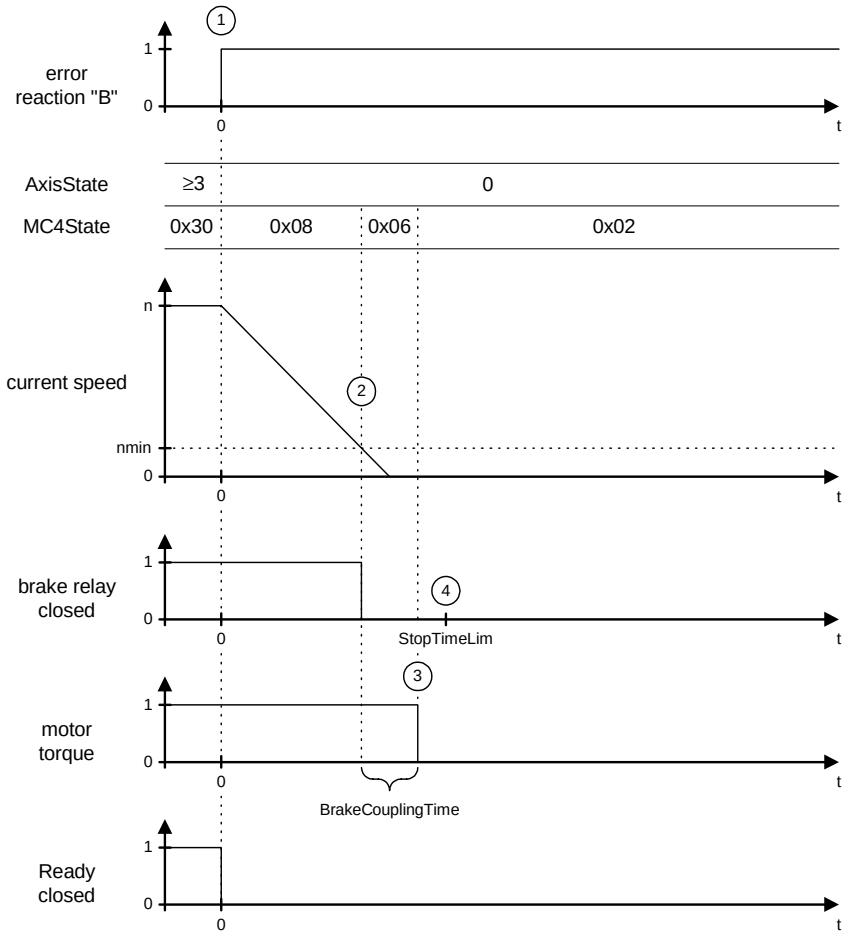



*Illustration 11-5: Diagramme des temps réaction d'erreur B / décélération avant expiration du temps de décélération*

Une erreur suivie de la réaction « B » (temps 1) donne lieu à une décélération avec un niveau de courant maximal et à l'ouverture du contact Ready. L'entraînement s'arrête avant l'expiration du temps de décélération maximal StopTimeLim (temps 4). Dès que la fréquence de rotation effective passe en-dessous du seuil inférieur (temps 2 : fréquence de rotation effective < nmin), le relais de frein s'ouvre. Le couple moteur est mis à zéro après l'expiration du délai d'enclenchement du frein *BrakeCouplingTime* (temps 3).
La décélération s'est donc effectuée sans erreur si la *CoastOption* pour la réaction d'erreur « B » n'est pas active (*bErrorCoast* = FALSE).

## 11.3.2 Dépassement du temps de décélération maximal

Une erreur avec la réaction d'erreur « B » donne lieu au comportement temporel suivant si la décélération ne s'achève pas avant l'expiration du délai de décélération maximal :

PDM_UserMan_ZeitDiag_fr.fm

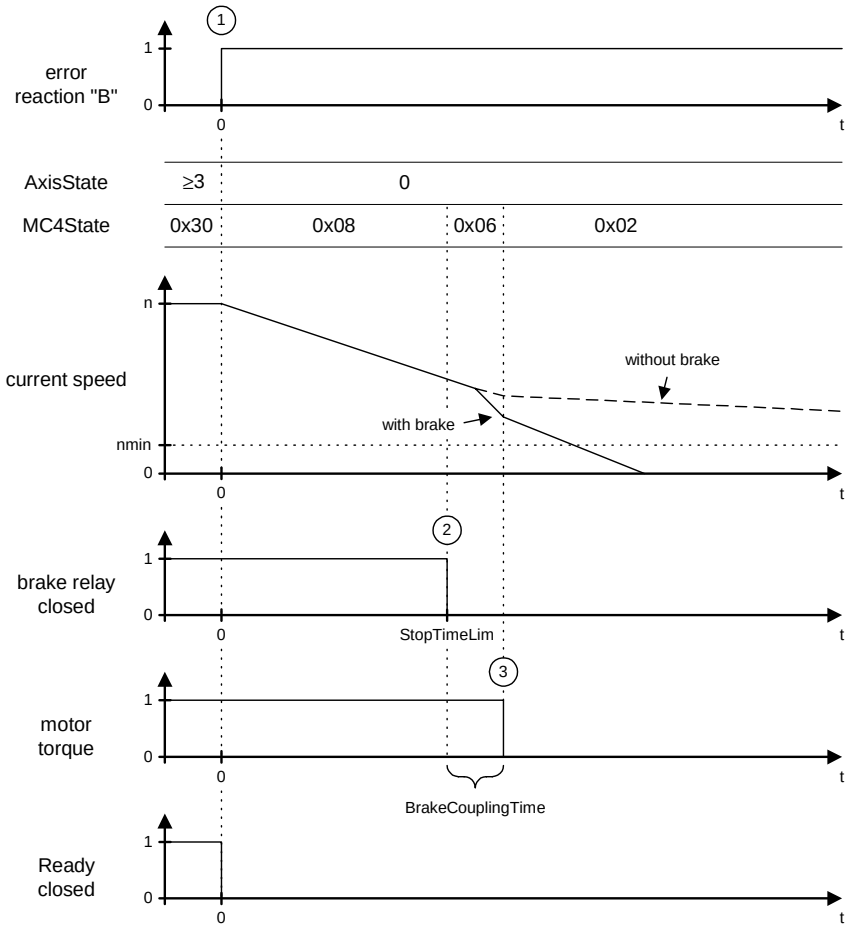



*Illustration 11-6: Diagramme des temps réaction d’erreur B /dépassement du temps de décélération maximal*

Une erreur suivie de la réaction « B » (temps 1) donne lieu à une décélération avec un niveau de courant maximal et à l’ouverture du contact Ready. Mais l’entraînement ne s’arrête pas avant l’expiration du temps de décélération maximal *StopTimeLim* (temps 2) (fréquence de rotation effective < nmin). Ce dépassement donne lieu à l’ouverture du relais de frein et à la transmission du message d’erreur « Stop time limit exceeded ». Mais ce message d’erreur reste sans effet sur l’entraînement parce qu’une erreur est déjà active avec la même réaction d’erreur. Le couple moteur est mis à zéro après l’expiration du délai d’enclenchement du frein (temps 3). L’entraînement est donc décéléré de façon réglée pendant un délai correspondant au double du temps de décélération maximal augmenté du délai d’enclenchement du frein, avant de s’arrêter de façon non réglée (par inertie). Mais ceci ne vaut qu’à condition que la *CoastOption* pour la réaction d’erreur « B » ne soit pas active (*bErrorCoast* = FALSE). L’arrêt non réglé de l’entraînement ne donne pas lieu à la génération d’un message d’erreur.
Le comportement d’un entraînement arrêté de façon non réglée

dépend, entre autres, de la présence d'un frein de maintien sur le moteur.

## 11.4 La réaction d'erreur « A »

Une erreur suivie de la réaction « A » entraîne le comportement temporel suivant :

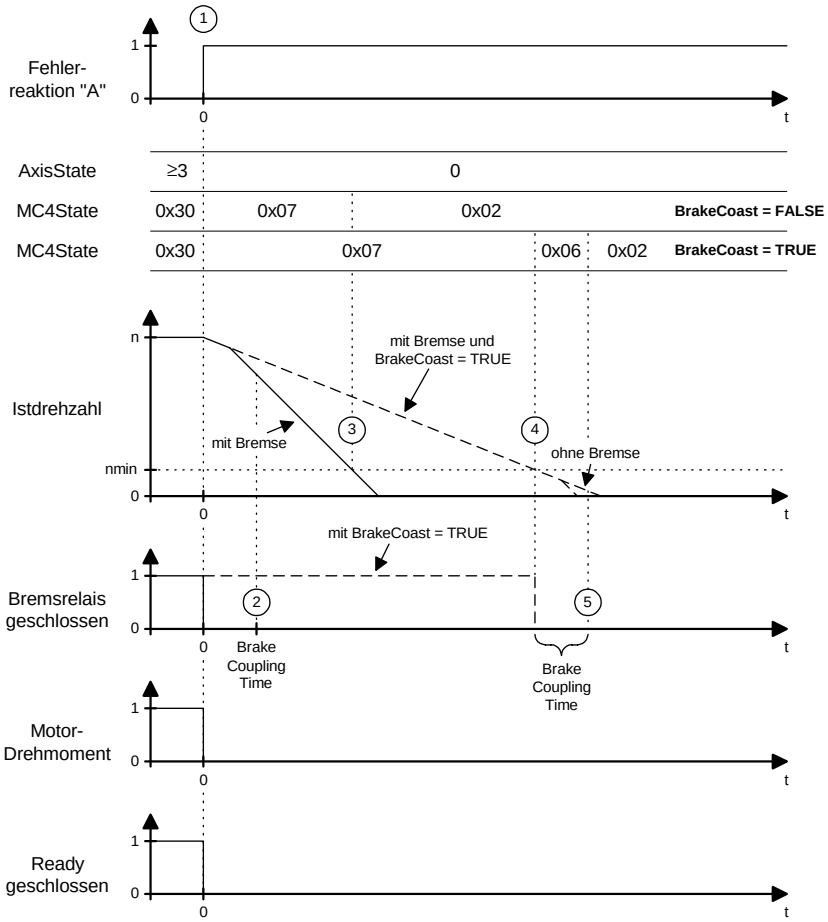



*Illustration 11-7: Diagramme des temps réaction d'erreur A*

Une erreur suivie de la réaction « A » (temps 1) provoque immédiatement la mise à zéro du couple moteur, l'enclenchement du frein et l'ouverture du contact Ready. Après l'expiration du délai d'enclenchement du frein *BrakeCouplingTime* (temps 2) et l'arrêt de l'entraînement (fréquence de rotation effective < nmin) (temps 3), MC4State passe à l'état 0x02. Après l'acquittement du message d'erreur, l'entraînement peut repasser de cet état au régime réglé. Avec la *CoastOption* pour le frein, celui-ci ne s'enclenche qu'au moment de l'arrêt (temps 4 : fréquence de rotation < nmin). MC4State repasse à l'état 0x02 après l'expiration du temps

PDM_UserMan_ZeitDiag_fr.fm

d’enclenchement du frein.
La même réaction se produit si la CoastOption pour la validation est activée (*bEnableCoast* = TRUE) au moment de l’arrêt de la validation ou si la *CoastOption* pour la réaction d’erreur « B » est activée (*bErrorCoast* = TRUE) au moment d’une erreur avec réaction d’erreur « B ».
Le comportement d’un entraînement arrêté de façon non réglée dépend, entre autres, de la présence d’un frein de maintien sur le moteur.

## 11.5 Arrêt par suite de surcharge (Overload)

L’arrêt par suite de surcharge peut en principe être suivi de deux séquences d’événements.

### 11.5.1 Arrêt par suite de surcharge avant expiration du temps d’arrêt maximal

L’arrêt par suite de surcharge donne lieu au comportement temporel suivant si la position de surcharge est atteinte avant l’expiration du temps d’arrêt maximal :

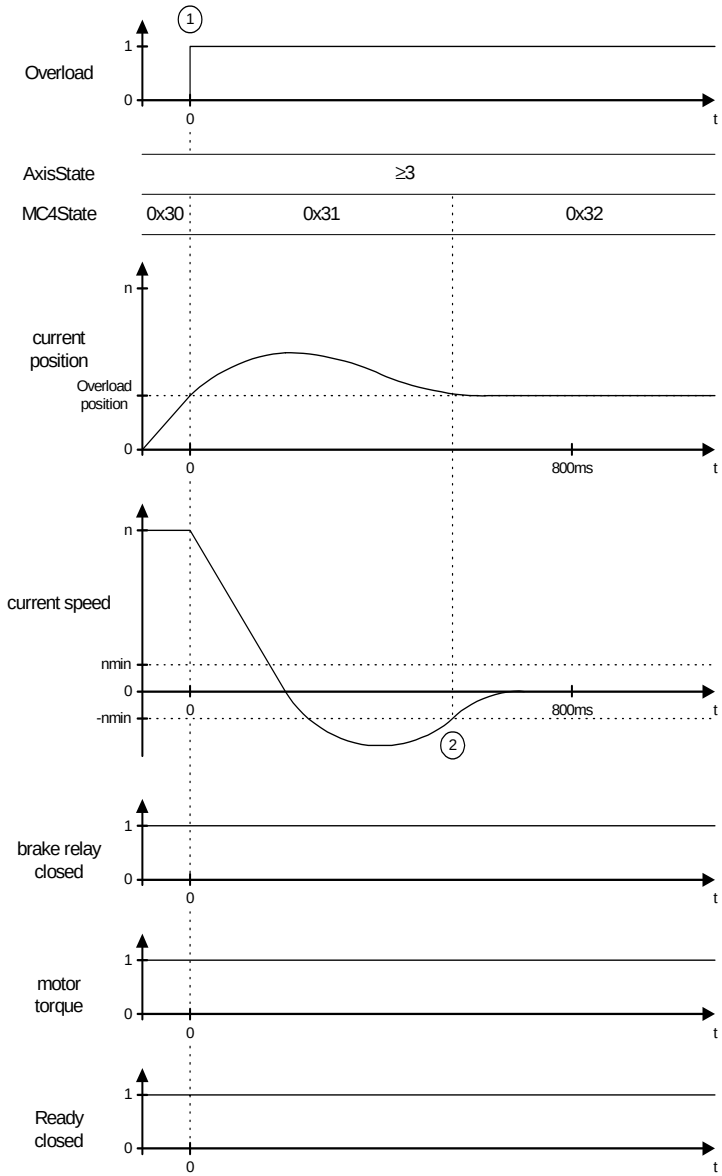



*Illustration 11-8: Diagramme des temps position de surcharge avant expiration du temps d'arrêt maximal*

Au moment de l'arrêt par suite de surcharge (temps 1), le réglage de l'entraînement passe à la position de surcharge. La position de surcharge est la position effective de l'entraînement au moment de l'arrêt par suite de surcharge. Si le réglage de l'entraînement atteint la position de surcharge avant l'expiration du temps d'arrêt maximal de 800 ms et si le moteur s'arrête (temps 2 : |fréquence de rotation effective| < nmin), MC4State passe à l'état 0x32. Le message de surcharge est ensuite acquitté, puis, le réglage de l'entraînement repasse aux consignes.
Cette procédure assure un arrêt par suite de surcharge exempt d'erreur.

PDM_UserMan_ZeitDiag_fr.fm

## 11.5.2 Arrêt par suite de surcharge avec dépassement du temps d'arrêt maximal

L'arrêt par suite de surcharge donne lieu au comportement temporel suivant si la position de surcharge n'est pas atteinte avant l'expiration du temps d'arrêt maximal :

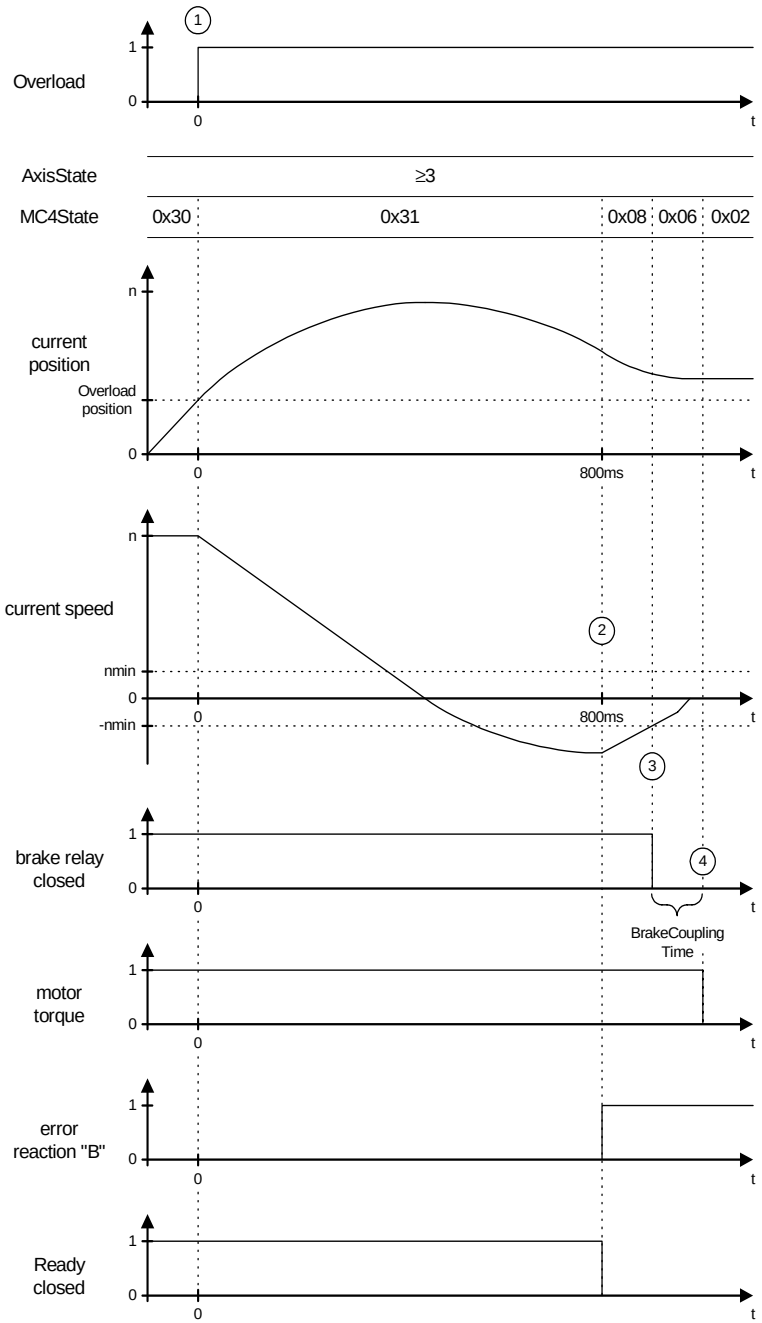



*Illustration 11-9: Diagramme des temps surcharge / dépassement du temps d'arrêt maximal*

Au moment de l’arrêt par suite de surcharge (temps 1), le réglage de l’entraînement passe à la position de surcharge. La position de surcharge est la position effective de l’entraînement au moment de l’arrêt par suite de surcharge. Si le réglage de l’entraînement n’atteint pas la position de surcharge avant l’expiration du temps d’arrêt maximal de 800 ms (temps 2), le message d’erreur « Stop time limit exceeded » avec réaction d’erreur « B » déclenche l’arrêt réglé de l’entraînement. Dès que la fréquence de rotation effective passe en-dessous du seuil inférieur (temps 3 : |fréquence de rotation effective| < nmin), le relais de frein s’ouvre. Le couple moteur est mis à zéro après l’expiration du délai d’enclenchement du frein *BrakeCouplingTime* (temps 4).
L’entraînement est donc arrêté si la *CoastOption* pour la réaction d’erreur « B » n’est pas active (*bErrorCoast* = FALSE).

**LEGENDE**

Les valeurs absolues sont indiquées directement sur l’axe. Elles se réfèrent au point origine de l’axe.

Les différences sont identifiées par des accolades. La parenthèse indique la plage de valeurs correspondante.

# 12 ENI-4

Ce chapitre résume la fonction, la structure et la commande de l’interface d’ingénierie ENI.

Documentation supplémentaire :

- Manuel d’utilisateur ENI Server
- Aide en ligne ENI Server, ENI Admin, ENI Control (composantes d’ENI Server Suite) et ENI Explorer
- Aide en ligne Automation Toolkit EPAS-4 (Atelier logiciel)

## 12.1 Fonction de l’interface d’ingénierie ENI

La complexité du monde de l’automatisation s’oppose à l’émergence d’un standard unique applicable à tous les produits matériels et logiciels. Les utilisateurs, d’autre part, refusent de voir leur choix limité à un petit nombre de constructeurs ; ils exigent, bien au contraire, de pouvoir faire appel à la solution qu’ils estiment la mieux adaptée à leurs besoins.

Pour offrir la transparence voulue, les systèmes d’automatisation ont désormais besoin d’interfaces ouvertes permettant un « arrimage » facile à l’environnement matériel et logiciel de l’utilisateur. Le système PacDrive comprend plusieurs interfaces de ce type. L’accès aux données du PacDrive Controller est ainsi possible à partir de serveurs OPC. L’interface la plus récente du système PacDrive est l’interface d’ingénierie ENI d’EPAS-4.

ENI est conçue comme système client/serveur. La partie serveur est un programme autonome faisant appel à un système de stockage de données, de préférence un système de gestion de versions (p.ex. Microsoft Visual SourceSafe ou MKS Source Integrity). Toutes les données d’ingénierie telles que blocs fonctionnels et listes de variables, mais également les visualisations et la configuration, sont stockées en mémoire via le serveur. Le format de stockage utilisé est XML. Cette interface met les données d’un projet réalisé avec EPAS-4 à la disposition d’autres outils. Elle leur permet notamment de lire, générer et modifier ces données.

Les trois exemples suivants permettront de mieux saisir le spectre des applications possibles et donc de mieux comprendre l’interface ENI :

- Mode multiutilisateur
- Gestion de versions
- Accès de clients externes

### 12.1.1 Mode multiutilisateur

ENI permet de réaliser des projets PacDrive en mode multiutilisateur. Les environnements de développement d'API courants ne permettent pas de réaliser des projets entre plusieurs développeurs. S'il était possible d'importer et d'exporter des blocs fonctionnels individuels, les développeurs devaient cependant s'entendre d'abord verbalement sur leurs actions.

L'interface ENI offre désormais une solution bien plus élégante à ce problème. Le projet complet est stocké dans l'interface. Pour intervenir sur un bloc fonctionnel, le développeur le charge dans son environnement de travail local. Ce bloc reste alors verrouillé pour les autres développeurs pendant toute la durée de l'intervention. Il sera à nouveau accessible après la validation de l'intervention par le développeur. Cette stratégie permet d'assurer un travail coordonné et donc plus efficace à plusieurs développeurs sur le même projet.

### 12.1.2 Gestion de versions

L'interface ENI gère l'utilisation d'un gestionnaire de versions de codes source. Le développement d'un logiciel suit généralement plusieurs étapes. Le programmeur réalisera donc les modules de son programme par étapes. Et il ne manquera pas de revenir sur certaines de ses modifications quand il constatera qu'elles ne répondent pas à ses attentes. Pour ce faire, il doit sauvegarder sa version et effectuer ensuite ses modifications sur une copie. Si les modifications s'avèrent efficaces, il continue de travailler avec cette copie, sinon il repasse à la version originale. Ceux qui pratiquent cette méthode savent qu'elle a besoin d'être améliorée.

Ces améliorations sont d'ores et déjà une réalité au niveau de l'environnement de développement EPAS-4 ainsi que ceux, très modernes, d'un grand nombre de langages de programmation complexes. Dans la banque de données de l'interface ENI, les développeurs peuvent désormais sauvegarder plusieurs versions de leur programme, mais également des blocs fonctionnels individuels. Le développeur peut appeler ces versions à tout moment. Pour que ce soit possible, il faut que l'interface ENI comporte une banque de données capable de gérer les versions. Parfaitement intégré dans l'environnement de développement EPAS-4, ce gestionnaire de versions n'entrave pas le déroulement des travaux.

### 12.1.3 Accès de clients externes

Très récemment encore, des clients externes ne pouvaient avoir accès aux données de l'environnement de développement. L'interface ENI permet désormais à tout client d'obtenir l'accès aux données d'un projet d'automatisation. Ces données peuvent être lues

et changées depuis le client externe. Une interface client en langage C++ (C++ DLL) permet de programmer des clients spécifiques au serveur ENI.

## 12.2 Structure et communications de l’interface ENI

ENI sauvegarde les objets des projets PacDrive dans un système de stockage de données. Il peut s’agir soit d’une base de données déjà existante soit d’un système de fichiers local. L’interface ENI comporte une partie serveur et une partie client de sorte que le système de stockage de données puisse être installé également sur d’autres calculateurs (condition préalable au mode multiutilisateur).

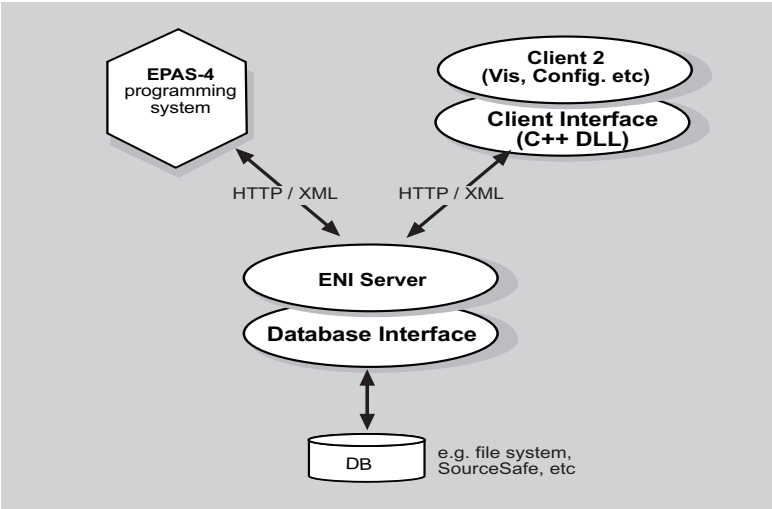



*Illustration 12-1: Le concept du serveur ENI*

### Le serveur ENI

Le serveur de l’interface d’ingénierie ENI est un service à part entière. Il communique en HTTP, les données utiles étant en XML.

### Le client ENI

EPAS-4, un système de programmation CoDeSys, mais également d’autres applications ayant besoin d’accéder à la base de données peuvent être les clients du serveur ENI. Tous les clients ont les mêmes droits.

### L’interconnexion entre le serveur ENI et la base de données (DB)

ENI ne définit pas son propre format de stockage. Des programmes pilotes (Database Interface) permettent de connecter le serveur ENI à un système de saisie de données existant. Sont actuellement disponibles les pilotes pour les bases de données
« Visual SourceSafe 6.0 », « MKS Source Integrity » et une base de

données locale. Pour l'interconnexion avec d'autres système de stokkage, veuillez nous contacter.

Le gestionnaire du serveur ENI détermine quel système de stockage de données celui-ci est en train de servir à un moment donné. (Voir également « ENI en tant que composante d'EPAS-4 » à la page 251.). Les clients du serveur auront alors accès au contenu de ce système.

**HTTP comme protocole de communication, XML comme format de données**

L'accès au serveur ENI se fait sous http, les données (objets) sont transmises en format XML, un standard très largement utilisé. Le protocole http permet de traverser des firewalls et offre des outils standards pour certaines opérations.

Une bibliothèque cliente C++ DLL (client interface) permet d'encapsuler le projet en classes C++, de sorte à ne pas limiter les logiciels d'accès au serveur ENI aux seuls formats http et XML.

## 12.3 Structure interne du système de stockage de données/L'explorateur ENI

Le serveur ENI gère les blocs fonctionnels générés par EPAS-4 et d'autres clients dans une structure de répertoires du système de stockage de données. Les blocs fonctionnels y sont sauvegardés sous forme d'objets identifiés par un type d'objet et des droits d'accès. De plus, un historique des versions y est tenu à jour pour chaque objet. A l'intérieur des répertoires, les objets doivent être clairement identifiables par leur nom et le type associé.

**L'explorateur ENI**

Comme l'explorateur Windows, l'explorateur ENI est un programme à part entière connecté au serveur ENI par l'intermédiaire des données d'accès de l'interface. L'explorateur ENI permet de visualiser la structure de répertoires et d'objets qui constituent l'interface ENI et d'appeler directement les fonctions de base de données à partir de cette structure, sans faire appel à EPAS-4 et quel que soit le système de saisie de données utilisé. L'explorateur ENI indique par ailleurs le type d'objet, les droits d'accès ainsi que l'état de contrôle momentané et l'identité de l'utilisateur.

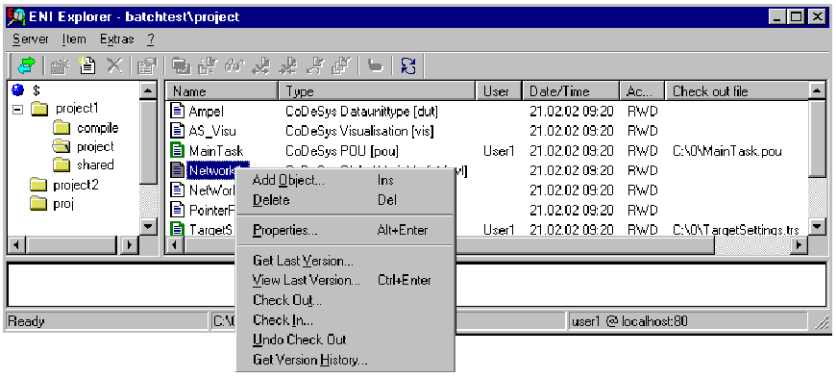



*Illustration 12-2: L’explorateur ENI*

## 12.4 ENI en tant que composante d’EPAS-4

Le système de programmation EPAS-4 est un client possible du serveur ENI. Chaque bloc fonctionnel d’un projet PacDrive peut être créé comme objet du système de saisie de données. Il est alors classé dans l’une des quatre catégories d’objet d’EPAS-4 (cf. ci-dessous). L’objet est mis en mémoire intégralement, c’est-à-dire que des caractéristiques telles que les droits d’accès en format XML sont transmises avec le contenu. Un échange automatisé avec le système de saisie de données peut être défini pour chaque catégorie d’objet.

### Le format XML destiné aux objets EPAS-4

L’exemple ci-dessous illustre la présentation d’un bloc fonctionnel en format XML. Il montre notamment : le nom du bloc fonctionnel (<name>), le type du bloc fonctionnel (<pou>), le chemin du bloc fonctionnel dans l’organisateur d’objets d’EPAS-4 (<path>), les droits d’accès pour les huit groupes de travail possibles (<accesslevels>) ainsi que le contenu du bloc fonctionnel dans la partie de déclaration (<interface>) et la partie de programme créée en l’occurrence en littéral structuré (<st>,<body>).

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<pou>
   <accesslevels> rw,rw,r,rw,r,rw,rw,rw </accesslevels>
   <path> \/languages </path>
   <name> ST_EXAMPLE </name>
   <flags> 4 </flags>
   <interface>
      <![CDATA[PROGRAM ST_EXAMPLE
      VAR
         YVAL: INT := -250;
         BOTTOM: INT := -250;
         RUN_STRING: STRING(20) := 'Start';
```

```
      OFFSET: INT := 2;
    END_VAR
  ]]>
  </interface>
  <st>
    <body>
      <![CDATA[RUN_STRING:='Start';
      IF (YVAL < 0) THEN
      YVAL := YVAL + OFFSET;
    BOTTOM := YVAL + OFFSET;
      END_IF;
    ]]>
  </body>
  </st>
</pou>
```

### Catégories d’objets ENI dans EPAS-4

L’interface ENI gère quatre types d’objets de données :

1. les objets spécifiques au projet : des blocs fonctionnels crées spécialement pour un projet déterminé
2. les objets communs : des blocs fonctionnels utilisés généralement par plusieurs projets, p.ex. des bibliothèques de blocs fonctionnels
3. les objets de compilation générés de façon automatique : ce type d’objet étant actuellement sans importance, l’interface ENI ne le gère pas
4. les objets locaux (non gérés par le système de stockage de données).

La classification d’un objet dans l’une des catégories se fait dans les options de projet et/ou les propriétés d’objet d’EPAS-4.

Les paramètres ENI peuvent être configurés séparément pour chacune des catégories 1 à 3 (cf. plus bas, Commande ...).

### Structure du projet PacDrive dans le système de stockage de données

Un ou plusieurs chemins sont définis pour le projet dans le système de stockage de données associé et donc « dans ENI » sous la racine configurée dans « ENI Control » ; ces chemins sont visualisés sous forme de répertoires contenant les blocs fonctionnels du projet (p.ex. un répertoire par catégorie d’objet ENI).

Chaque objet du projet contient un type de données Les noms des blocs fonctionnels repris d’EPAS-4 reçoivent donc une extension correspondant au type de données ainsi qu’un symbole de type, ce qui leur permet de représenter les objets à l’extérieur du système de programmation.

### Commande de l'interface ENI dans le système de programmation

ENI peut être activée en tant qu'option de projet dans EPAS-4. Des boîtes de dialogue sont alors disponibles pour les fonctions suivantes :

1. Dans les options de projet, identification de la catégorie d'objets ENI dont relèvera un objet nouvellement créé.

2. Configuration du type de connexion vers ENI , dans les options de projet, pour chacune des trois catégories d'objets ENI (objets de projet, objets communs, fichiers de compilation ; voir plus haut) :

- Données d'accès : Port ; droits d'accès ; projet de la zone mémoire ENI auquel le projet actuel doit être associé)
- Spécification des temps et des modalités de l'échange automatique avec le système de stockage de données (p.ex. appel des objets dans la base de données à chaque ouverture d'un projet ou sortie au début d'une séance de modification sous EPAS-4, etc.)

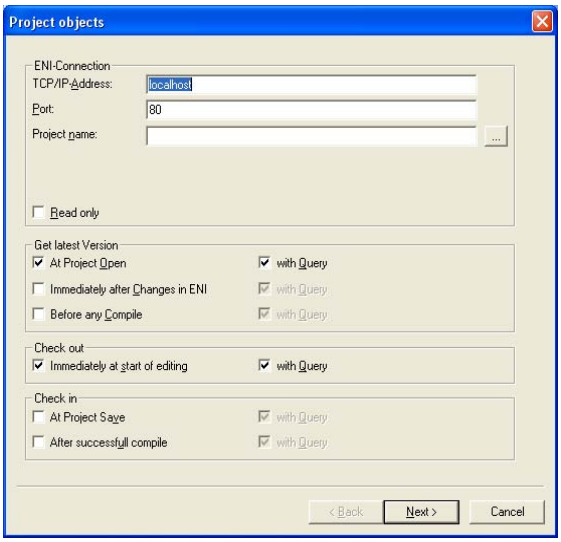



*Illustration 12-3: Réglages ENI*

3. Classification individuelle d'un objet dans une catégorie d'objets (voir plus haut) dans les propriétés de chaque objet

4. Saisie du nom d'utilisateur et du mot de passe dans une boîte de dialogue d'identification (cf. menu gestion de projet) de la base de données pour l'accès au système de saisie de données

5. Commandes de base de données, dans le menu de gestion de projet, pour des objets individuels ou le projet intégral :

- Call / call all (appeler / appeler tous)
- Check out / check out all (sortir / sortir tous)
- (Check in / check in all (rentrer / rentrer tous)

- Undo check out / … for all objects (annuler sortie / ... pour tous les objets)

- Version history for ofject / project (historique des versions de l'objet et/ou du projet)

- Show changes (afficher les modifications)

- Label version (marquer les versions)

-Insert common projects (insérer des objets communs) (reprise dans le projet local depuis le système de base de données)

- Update status (rafraîchir l'état)

## 12.5 ENI Admin (Gestionnaire ENI) et ENI Control (Contrôleur ENI)

Certains réglages ne sont possibles que localement au niveau du serveur ENI. Les programmes « ENI Admin » et « ENI Control » permettent de procéder à ces réglages. Comme le serveur et l'explorateur ENI, ces deux programmes sont installés en tant que composantes de l'« ENI Server Suite » représentée par un icône dans la barre système de votre PC.

**Pour lancer « ENI Admin »** (ENIAdmin.exe), il faut entrer le mot de passe de l'administrateur. Ce programme permet de définir les utilisateurs et groupes d'utilisateurs, les droits d'accès et le mot de passe de l'administrateur, il offre des fonctions de gestion de la licence et de visualisation des types d'objet de données.

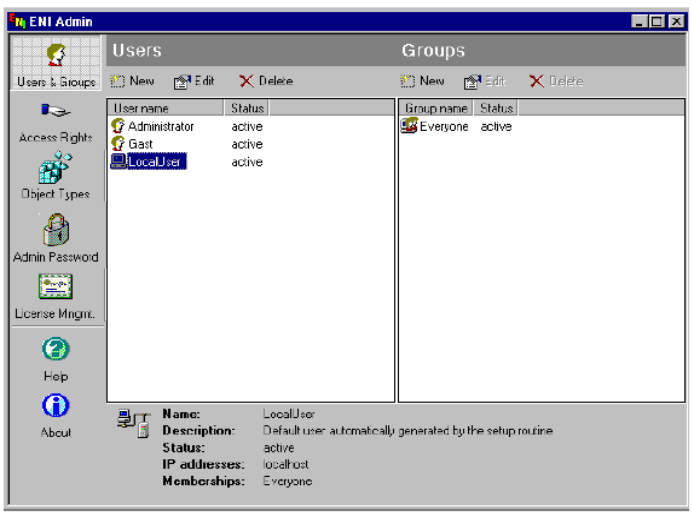



*Illustration 12-4: ENI Admin*

**De même que pour « ENI Admin », le lancement d'« ENI Control »** (ENIService.exe) nécessite la saisie du mot de passe de l'administrateur ENI.  « ENI Control » permet de choisir le pilote de la base de

données, c'est pourquoi le programme d'installation l'utilise au cours de la procédure d'installation. Le contrôleur ENI offre par ailleurs la possibilité d'inactiver et de relancer l'interface ENI, et de l'associer à un autre compte utilisateur. Et ce programme permet de modifier certains paramètres de communication (temporisateur de communication, port, etc.) de la connexion client-serveur. L'enregistrement des événements (Event Log) est également prévu.

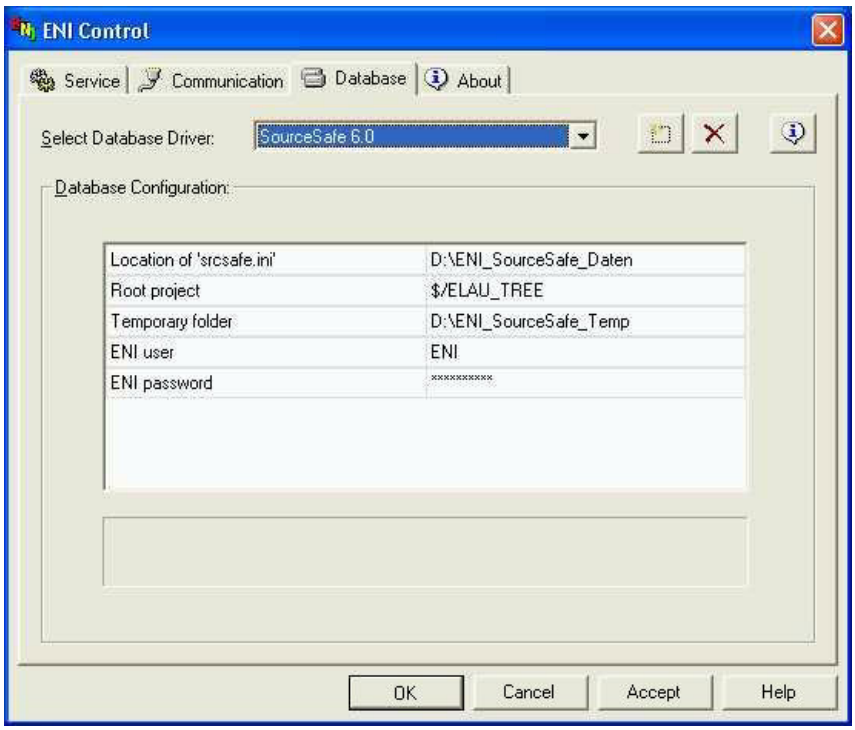



*Illustration 12-5: ENI Control*

# 13 FAQ (Foire Aux Questions)

**C'est quoi, les variables VAR_IN_OUT ?**

Les variables VAR_IN_OUT sont transmises en tant que pointeurs. Il n'est donc pas nécessaire de copier des données vers ou à partir d'une instance.

**Y a-t-il des pointeurs ?**

La norme IEC 61131 ne connaît pas des pointeurs. Le système PacDrive supporte pourtant des pointeurs. En cas d'utilisation de pointeurs, le déroulement correct des logiciels ne peut pas être garanti - c'est l'utilisateur qui assume la responsabilité lui-même.

**Y a-t-il des possibilités de structuration pour la portée des variables ?**

Il est possible de créer des variables locales et globales.

**Quel langage peut être utilisé pour quel but ?**

Il n'y a pas une marche à suivre qui s'impose. Recommandation :

- AS pour des séquences en étapes
- ST pour des calculs algorithmiques
- FUP pour des appels de programme
- KOP pour des verrouillages, par exemple
- AWS pour le traitement de bits

**Quel langage de programmation est le plus rapide ?**

En ce qui concerne la vitesse d'exécution, il n'y a que des différences peu importantes entre les langages de programmation.

**Y a-t-il une fonction oscilloscope ?**

Oui, l'enregistrement de trace.

**Comment afficher une instance d'un bloc fonctionnel (FB) en mode en ligne ?**

Il est possible d'afficher une instance d'un bloc fonctionnel sous PROJECTS / OPEN INSTANCE. Pour faire cela, le curseur dans la fenêtre Object Organizer (fenêtre gauche) doit être positionné sur le bloc fonctionnel ou bien le FB doit être ouvert déjà.

**Y a-t-il des compilations partielles ou faut-il toujours compiler le programme entier ?**

Avec PROJECT / COMPILE ALL, le programme entier est recompilé. En cas d'une modification en ligne (Online-Change), seulement le code modifié est généré et chargé. Après avoir sélectionné Rebuild all, une modification en ligne n'est plus

possible. Il faut entièrement recharger le programme et effectuer une reprise à froid.

**Que se passe-t-il au cours de modifications en ligne (Online-Change) ?**

Ce procédé permet des modifications avec l'automate programmable en cours de fonctionnement. Les modifications en ligne (Online-Change) permettent de créer le code Delta, c.à.d. les dernières modifications. Directement après la procédure d'identification, ce code est transmis à la commande (le système de traitement) et y sauvegardé. Lors de l'activation des modifications, le code est ajouté à la fin du cycle en cours. Puisque le code existe deux fois alors, une réorganisation du mémoire a lieu au cours des cycles suivants.

**C'est quoi, OPC ?**

OPC signifie OLE for Process Control et représente une interface standardisée pour la connexion des composants matériel les plus divers aux applications HMI/SCADA.
Pour des informations plus détaillées voir le Guide utilisateur Serveur OPC (numéro d'article 17130073-00x).

# 14 Glossaire

**Type de donnée dérivé**

Les types de donnée spécifiques à l'utilisateur sont générés à l'aide de définitions de type. Ces types de données spécifiques se composent de types de données élémentaires et/ou dérivés.

**Diagramme fonctionnel en séquence (SFC)**

Diagramme fonctionnel en séquence (angl.: Sequential Function Chart - SFC) est un langage de programmation qui décrit des procédures de commande séquentielles ou parallèles avec gestion du temps et des événements.

**Résistance de terminaison**

Résistance pour ajuster la puissance sur le bus ; les résistances de terminaison sont généralement installées aux extrémités des câbles de bus et leurs segments.

**Codeur absolu**

Chaque point du parcours est identifié par un signal univoque. Les systèmes à codeurs absolus ne nécessitent aucune course de référence après la mise sous tension.
L'information du codeur est lue à l'aide d'un disque codeur supplémentaire capable de transmettre jusqu'à 4096 rotations. Une variante plus simple est la sauvegarde des informations à l'aide d'une pile intégrée.

**Fréquence d'échantillonnage**

Intervalle en millisecondes pour le calcul des valeurs de consigne d'un axe, la comparaison de ces consignes avec les valeurs effectives et le traitement des différences par le régulateur.

**Identification d'axe**

Adresse logique d'un MotorController MC-4 (il ne s'agit pas de l'adresse RealTimeBusAdr !)

**ActiveX**

*ActiveX* est un nom collectif. Il s'applique à toutes les technologies regroupées sous le terme OLE Control (*OCX*). Mais le sens respectif de chacun de ces termes a évolué depuis la définition d'*ActiveX*. *ActiveX* est une technologie Microsoft de gestion de contenus interactifs dans le World Wide Web. Pour entrer dans la catégorie des commandes *ActiveX*, un *objet COM* doit offrir certaines *interfaces*.

PDM_UserMan_Glossar_fr.fm

**Signal analogique**

De nombreuses grandeurs de procédure sont transmises sous forme analogique. On distingue entre les signaux de tension et les signaux de courant. Pour lire une grandeur analogique, l'ordinateur doit être équipé d'un module convertisseur analogique-numérique. La résolution des signaux dépend de la précision du convertisseur AN.

**Bus des entraînements**

Les commandes des systèmes d'automatisation les plus avancés communiquent avec leurs entraînements en mode numérique. Le bus des entraînements doit avant tout assurer une transmission rigoureusement déterministe des données de mouvement.

**Liste d'instructions (IL)**

Le langage IL (angl.: Instruction List - IL) est un langage de programmation très répandu pour API ; il s'apparente à un langage d'assembleur.

**Applet**

Un applet est un programme utilisé essentiellement sur les pages web. Ecrit en Java (langage de programmation), il peut être exécuté sur toutes les plate-formes.

**ASCII**

Abréviation d'American Standard Code for Information Interchange. Les signes en code ASCII ont une longueur de 7 bits.

**AS Interface (ASI)**

L'AS Interface (ASI) est d'une solution d'interconnexion particulièrement simple entre systèmes d'automatisation. Elle est optimisée pour le niveau de terrain le plus bas et permet de raccorder des modules esclaves binaires et intelligents ainsi que des équipements plus simples travaillant en mode analogique. L'AS Interface ne peut concurrencer les bus de terrain du point de vue des volumes de données transmises.

**Hiérarchie des appels**

Cette fonction permet de visualiser la pile des appels de l'enregistreur des messages pour un message de diagnostic en attente d'acquittement.

**Baud**

Vitesse de transmission de données définie par le nombre de bits transportés par seconde.

**Débit de bauds**

Le débit en bauds correspond à la vitesse de transmission des données ; il indique le nombre des bits transmis par seconde (débit de bauds = débit binaire).

**Bloc**

Voir également « UOP » à la page 275.

**Temps d’accélération**

Temps d’accélération permettant à un ensemble servomoteur-amplificateur non chargé de passer d'une vitesse zéro à une vitesse nominale avec couple crête.

**Potentiel de référence**

Potentiel par rapport auquel les tensions d'un circuits électrique sont prises en considération et/ou mesurées.

**Amorçage**

On appelle amorçage la procédure de lancement d’un « calculateur » (p.ex. PC) et de son logiciel d’exploitation.

**Navigateur**

Un navigateur web (web browser) est un programme permettant d’accéder aux pages web et de les afficher. Il a pour fonction essentielle le téléchargement et l’affichage de documents HTML et des images associées.

**Bus**

Support de transmission commun reliant tous les utilisateurs et possédant deux extrémités définies.

**Connecteur de raccordement de bus**

Connexion physique entre une station et le câble de bus.

**Segment de bus**

Voir également « Segment » à la page 279.

**Système de bus**

Ensemble des stations reliées physiquement par un câble de bus.

**CAN**

Abréviation de Controller Area Network, un système de bus pour véhicules développé par Bosch et Intel. Entre-temps, CAN a fait ses preuves en tant que bus de terrain pour systèmes d’automatisation. Comme la quasi-totalité des bus de terrain, CAN utilise le modèle OSI à 7 couches. La norme ISO 11898 n’est applicable à CAN que pour les couches OSI 1 et 2 ; CAN est basé

sur une structure de bus. La vitesse de transmission peut varier entre 20 kBit/s et 1 Mbit/s. La couche d'application est définie pas des protocoles de couche 7 comme DeviceNet ou CANopen, spécialement adaptés à CAN. Grâce à sa procédure d'accès CSMA/CA et son système d'adressage orienté objet, CAN permet de réaliser des systèmes performants à commande par événements. Système multimaître, CAN permet d'échanger des données entre un nombre illimité d'utilisateurs.

**Marquage CE**

Le marquage CE certifie la conformité d'un produit à l'ensemble des directives européennes applicables.

**Client**

Application utilisant un serveur pour lui faire exécuter certaines tâches.

**CLSID (pour OPC)**

Code univoque mondialement répandu pour l'identification d'*objets COM*. *CLSID* permet à un client de s'adresser à un *objet COM* défini. Un code *CLSID* est composé de 128 bits. Le caractère univoque des codes *CLSID* est préservé par l'insertion de la date et de l'heure de sa création et d'un descriptif de la configuration matérielle de l'objet.

**Commande numérique (CNC)**

Abréviation de Computerized Numerical Control. Les commandes numériques (CNC) équipent principalement les machines outils.

**COM**

Le **C**omponent **O**bject **M**odel de Microsoft définit un mécanisme de collaboration entre composantes logicielles de systèmes d'exploitation Microsoft Windows. *COM* permet à une application d'utiliser les services d'autres applications ou composantes. *COM* est la technologie de base pour *ActiveX*, *OLE* et *DCOM*. Définissant la collaboration des composantes logicielles au bit près, *COM* ne dépend pas, en principe, d'un langage de programmation ou d'un système d'exploitation particulier. Le standard COM est désormais disponible sur d'autres systèmes d'exploitation.

**Objets COM**

Les *objets COM* sont des composantes logicielles conformes aux modèle *COM* et dédiées à un *client* qui utilise leurs fonctionnalités par l'intermédiaire d'interfaces COM. Les objets COM sont définis de façon univoque par un code *CLSID*.

**ControlNet**

Réseau de commande ouvert, ControlNet offre un débit binaire élevé particulièrement intéressant pour les applications à temps réel. ControlNet assure l'interconnexion de commandes, modules E/S, PC, systèmes de visualisation (interfaces homme-machine) et autres équipements intelligents. Avec sa vitesse de transmission de 5 Mbit/s, ControlNet autorise une transmission déterministe et répétable de données de commande à durée critique. Un algorithme CTDMA (Concurrent Time Domain Multiple Access) à tranches de temps commande l'accès au réseau.

**Continuous Function Chart (CFC)**

Continuous Function Chart (CFC) est un langage à éléments booléens et arithmétiques activés simultanément ; CFC présente des points communs avec FBD. Les programmes CFC ne sont pas réalisés sous forme de réseau. Leurs éléments peuvent être placés librement. Les listes de programmation sont composées d'éléments différents tels que bloc, entrée, sortie, saut, étiquette, retour et commentaire. Les entrées et sorties de ces éléments sont reliées par simple mouvement de la souris. La ligne de connexion se trace de façon automatique.

L'éditeur CFC se distingue avantageusement de l'éditeur FBD courant par la possibilité d'insertion directe d'asservissements.

**UC**

Unité de traitement centrale, microprocesseur, unité centrale

**CUL**

Les produits portant le marquage CUL sont conformes aux normes de sécurité canadiennes. Les normes de sécurité canadiennes peuvent différer des normes UL en vigueur aux Etats-Unis.

**Cv**

Identificateur de la vitesse normalisée -> cf. VDI 2143

Signification :

Cv inférieur -> le couple d'entraînement nécessaire est inférieur en raison de la sollicitation statique (ressort, inertie, force utile).

**DCOM**

**D**istributed **C**omponents **O**bject **M**odel est une extension du standard COM permettant d'utiliser des *objets COM* entre plusieurs calculateurs. *DCOM* est un protocole optimisé permettant à un client d'utiliser un *objet COM* sur un autre calculateur. Pour ce faire, les appels adressés à l'*objet COM* sont transformés en paquets de données, acheminés via le réseau, restitués et présentés à l'*objet COM*. L'utilisation d'objets distants est transparente pour le *client*.

**Déclaration**

Indication de variables et d'instances de blocs fonctionnels dans un bloc de déclaration comprenant notamment un identificateur, le type de données ou de blocs fonctionnels ainsi que, le cas échéant, les valeurs initiales, les domaines de valeur et les caractéristiques du terrain.
Ce terme désigne par ailleurs la définition et/ou la programmation d'UOP, étant donné que celles-ci sont « déclarées » au système de programmation avec leurs caractéristiques.

**Bloc de déclaration**

Regroupement de *déclarations* relatives à un même type de variables en début de l'*UOP*.

**DeviceNet**

DeviceNet est un standard de bus de terrain ouvert non propriétaire pour systèmes d'automatisation. DeviceNet permet d'interconnecter les API avec des équipements « intelligents » tels que capteurs, entraînements, modules E/S, auxiliaires de commande manuels et interfaces homme-machine de conception simple via un câble unique. DeviceNet est un protocole d'application (couche OSI 7) basé sur le standard de réseau CAN (Controller Area Network). Sa fiabilité excellente le destine aux applications sophistiquées aux très nombreux modules E/S. Suivant le type et la longueur du câble de bus, sa vitesse de transmission varie entre 125 kBit/s et 500 kBit/s.

**Technologie décentralisée**

Dans le domaine des changeurs de fréquence et des survolteurs-dévolteurs, on distingue entre solutions centralisées et décentralisées. La différence se situe au niveau du mode d'alimentation en tension. Une alimentation centralisée est principalement constituée d'un module qui génère les tensions destinées aux différents amplificateurs. L'alimentation décentralisée se compose de blocs d'alimentation individuels pour chaque amplificateur ou drive
« MotorController ».

**Connexion à distance**

La connexion à distance (pour la transmission de données à distance) désigne tous les transferts de données entre équipements ou systèmes informatiques sur de longues distances. Les supports de transmission ou média de transmission sont : câbles électriques, câbles à fibre optique, radio.

**Variable directement représentée**

*Variable* correspondant à une *adresse hiérarchique*, mais sans autre identificateur.

**Constante de couple**

Quotient du couple d’arrêt et du courant d’arrêt exprimé en Nm/A.

**Régulateur de vitesse**

Le régulateur de vitesse ajuste la différence entre consigne de vitesse et vitesse effective à un niveau près de zéro. Sortie : Consigne de courant.

**Système d’exploitation à temps réel**

Comme c’est le cas des systèmes d’exploitation classiques, les systèmes d’exploitation à temps réel sont conçus pour l’exécution en parallèle de tâches, files ou procédures. Pour remplir le critère du temps réel de l’API, les systèmes d’exploitation à temps réel utilisent des méthodes d’attribution de temps de calcul spécifiques. La prédictibilité (déterminisme) de leur comportement, c'est-à-dire la certitude qu’une tâche déterminée sera effectuée dans un délai défini préalablement, est l’une des principales caractéristiques des systèmes d’exploitation à temps réel. Les systèmes d’exploitation à temps réel doivent par ailleurs répondre à des critères particuliers de gestion de ressources (zones mémoire communes et exclusives, modules E/S, ports ou services de temps) et de communication de tâches via canal sémaphore, boîte aux lettres et files d’attente d’événements.

**Procédure temps réel**

La procédure temps réel comprend, entre autres, le traitement des codeurs maîtres et des codeurs logiques, l’exécution des ordres de mouvement, avec notamment le calcul des consignes, la gestion des entrées de mesure, la saisie des valeurs de trace, sans oublier l’échange de données temps réel via le bus temps réel (SERCOS).

**Système en temps réel**

Une norme DIN désigne par ce terme un système capable de toujours réagir à un événement extérieur de façon définie et de générer sa réponse dans un délai spécifique (temps d’échantillonnage). Le non-respect de ce délai est interprété comme une défaillance du système intégral. S’il s’agit d’une procédure à criticité inférieure, un dépassement est toléré dans certaines conditions. On parle alors de « temps réel mou ». Un système en temps réel mou est acceptable si le dépassement des délais imposés n’entraîne aucun danger de dégât matériel ou humain. Les machines d’emballage requièrent en général des API à temps réel « dur ».

**Variable à un seul élément**

Variable basée sur un seul type de données.

**Machine d’emballage électronique**

Dans une machine d’emballage électronique, l’arbre de renvoi mécanique est remplacé par un arbre de renvoi électronique à entraînements électriques individuels d’une grande flexibilité.

**Type de donnée élémentaire**

Un *type de donnée* standard défini par la norme IEC 61131-3.

**Compatibilité électromagnétique (CEM)**

Le sigle CEM (compatibilité électromagnétique) désigne l’aptitude d’un équipement électrique ou électronique à fonctionner de façon satisfaisante par rapport à son environnement électromagnétique. Les valeurs limites pour l’émission de perturbations et l’immunité aux perturbations sont définies par des normes.

**EN**

Abréviation pour « norme européenne ». Le remplacement des normes nationales (DIN, VDE ...) par des normes européennes est en cours.

**Interrupteur de fin de course**

Interrupteur limitant le déplacement d’un élément de machine.

**Prise de terre**

Un ou plusieurs élément(s) conducteur(s) en contact direct avec la terre.

**Mise à la terre**

La mise à la terre consiste à relier un composant conducteur avec la prise de terre.

**Structure non reliée à la terre**

Structure dépourvue de liaison galvanique avec la terre. Les courants parasites sont généralement supprimés à l’aide d’une combinaison résistance-capacité.

**Défaut à la terre**

Liaison électrique entre une phase et le conducteur de protection.

**Ethernet**

La technologie Ethernet existe depuis le début des années 80 et s’est imposée dans le monde des technologies de réseau. Elle gardera certainement pour longtemps encore cette place prépondérante. Ethernet utilise différents types de câble, des fibres optiques et la transmission radio ; le débit va de 10 Mbits/s à 1 Gigabit/s. Les équipements de tous types et de toutes générations travaillent en réseau sans problème. L’unique inconvénient

d’Ethernet, qui est de ne permettre qu’une communication déterministe et réservée, n’a pas réellement d’incidence pour de nombreuses applications (interfaces homme-machine par exemple) ; il est même insignifiant étant donné les capacités de transmission quasi-illimitées.

**FBD**

Ce langage en blocs fonctionnels correspond pour l’essentiel au diagramme de blocs fonctionnels classique.

**Tableau**

Enchaînement d’éléments du même *type de données*.

**Bus de terrain**

En règle générale, les capteurs et actuateurs communiquent avec l’API ou une unité de traitement de données par une connexion analogique ou numérique. Chacune de ces connexions entre capteur ou actuateur, d’une part, et l’API, d’autre part, nécessite alors un câble à plusieurs conducteurs. Au niveau de l’automate programmable, un circuit d’entrée et/ou de sortie (E/S) est nécessaire pour chacun de ces éléments périphériques Un bus de terrain permet de regrouper toutes ces lignes de raccordement dans un seul câble de bus. Les entrées et sorties sont remplacées par un module d’interface. Cette solution présente un certain nombre d’avantages évidents. Plus facile à mettre en service et à entretenir, le système d’automatisation devient également plus flexible. Ces atouts se traduisent en règle générale par des avantages de coût par rapport aux systèmes classiques. La diversité des cahiers des charges, le grand nombre de technologies ainsi que les politiques divergentes des acteurs du marché ont empêché l’émergence d’un standard unique et favorisé la démultiplication des systèmes de bus de terrain. Les systèmes d’ELAU AG gèrent plusieurs standards de bus internationaux : CANopen, PROFIBUS DP, DeviceNet et ControlNet.

**Front**

Un front « montant » correspond au passage d’une variable booléenne de 0 -> 1. Un front « descendant » représente, par conséquent, le passage de 1 -> 0.

**« Programmation par blocs semblable au système distribué »**

cf. Continuous Function Chart (CFC).

**FTP**

FTP (pour « file transfer protocol », protocole de transfert de données) est un protocole de transmission utilisé dans le World Wide Web ou dans des réseaux locaux communiquant sous TCP/IP. Le « FTP anonyme » est un service proposé par quelques

milliers de serveurs internet ; il permet de télécharger des fichiers sans indication de mot de passe.

**Diagramme fonctionnel (DF)**

Le diagramme fonctionnel est un langage de programmation graphique. Il utilise une liste de réseaux dont chacun représente, par sa structure interne, un terme logique et/ou arithmétique, l'appel de blocs fonctionnels, un saut ou bien une instruction de retour.

**Langage en blocs fonctionnels (FBD)**

Le langage en blocs fonctionnels (angl.: Function Block Diagram - FBD) est un langage de programmation conçu pour décrire des réseaux d'éléments simultanément actifs, de type booléen, arithmétique ou autre.

**global**

intégral ; ici : applicable à la configuration dans son ensemble.

**GSM**

GSM est un standard européen de téléphonie mobile dans la bande de 900 MHz ; ce standard a été défini par la Conférence Européenne des Administrations des Postes et Télécommunications (CEPT). Grâce aux services de données GSM qui font appel à des technologies de transmission numériques, les machines d'emballage bénéficient désormais du télédiagnostic via le téléphone portable.

**Frein de maintien**

Quand le système est actif, le frein de maintien ne doit être serré qu'après l'arrêt complet du moteur.

**Adresse hiérarchique**

Adresse physique du poste d'enfichage du module d'entrée/sortie d'un API.

**Interface homme-machine (HMI)**

Les interfaces homme-machine (Human Machine Interface) équipent des systèmes de fabrication industrielle dont elles permettent la commande, la surveillance ainsi que la sauvegarde des données. Si les machines d'emballage de taille réduite utilisent des appareils de commande opérateur relativement simples, les chaînes d'emballage plus complexes sont quant à elles pilotées par des dispositifs de visualisation performants installés sur des PC industriels. Ces dispositifs affichent les modes opératoires successifs d'une machine sous forme de diagrammes fonctionnels qui facilitent le contrôle des machines en rendant leur fonctionnement plus transparent. Les principaux paramètres

opérationnels sont enregistrés et figurent dans les diagrammes. Toute anomalie de fonctionnement est immédiatement signalée par une alarme.

**HTML**

HTML signifie HyperText Markup Language. Il s'agit donc d'un langage de marquage (Markup Language) conçu pour décrire les éléments logiques d'un document. Le langage HTML est notamment constitué de commandes de marquage des éléments caractéristiques d'un document (titres, paragraphes, listes, tableaux, références graphiques).

**Id**

cf. Identification

**IEC 61131**

IEC 61131-3 est un standard international pour les langages de programmation des automates programmables industriels (API). Les langages relevant de cette norme sont la liste d'instructions (IL), le diagramme fonctionnel (DF), le langage à contacts (LD), le littéral structuré (ST) et le diagramme fonctionnel en séquence (SFC).

**Appel indirect d'un bloc fonctionnel**

Appel d'une instance de blocs fonctionnels dont le nom a été transmis à *UOP* en tant que paramètre VAR_IN_OUT.

**Codeur (rotatif) incrémental**

Codeur transmettant sa position par deux signaux décalés de 90°. Les positions transmises ne sont pas des positions absolues.

**Instance**

Ensemble des données d'un bloc fonctionnel ; l'instance est structurée par une *déclaration* qui spécifie notamment le type du bloc.

**Instanciation**

L'instanciation consiste dans l'association de variables, avec indication notamment de leur nom et du type de données, à un bloc fonctionnel dans le cadre de la déclaration de celui-ci par l'utilisateur. L'association de plusieurs types de variables à un seul type de bloc fonctionnel entraîne la création d'autant d'instances dont chacune donnera lieu à la sauvegarde d'une copie du bloc fonctionnel dans la mémoire de l'API.

**Interface**

Interface

**Internet**

L’Internet est un réseau décentralisé, c'est-à-dire qu’il ne dépend d’aucun ordinateur en particulier. Ce réseau est constitué d’un certain nombre de sous-réseaux ; le protocole utilisé est TCP/IP.

Principaux services offerts par l’internet:

- courrier électronique – le service le plus fréquemment utilisé
- World Wide Web – second service le plus fréquemment utilisé.
- Usenet ou groupes de discussion
- FTP ou service de transfert de fichiers.

**Inverter Enable**

Le commutateur Inverter Enable commande le mode d’arrêt sécurisé (safe stop) du MotorController MC-4. Ce mode permet d’exclure qu’un entraînement arrêté se remette en marche de façon intempestive.

**IP 20**

Degré de protection spécifié dans la norme DIN 40050 : Protection contre les contacts directs (doigts) et la pénétration de corps solides étrangers d’un diamètre de 12 mm ou plus.

**IP 65**

Degré de protection spécifié dans la norme DIN 40050 : Protection totale contre les contacts directs. Etanchéité à la poussière et protection contre les jets d’eau de toutes directions à la lance.

**IP 66**

Degré de protection spécifié dans la norme DIN 40050 : Protection totale contre les contacts directs. Etanchéité à la poussière et protection contre les projections d’eau assimilables aux paquets de mer.

**IP 67**

Degré de protection spécifié dans la norme DIN 40050 : Protection totale contre les contacts directs. Etanchéité à la poussière et protection contre les effets nuisibles d’une immersion temporaire dans l’eau dans des conditions normalisées de pression.

**RNIS**

RNIS (Réseau Numérique à Intégration de Services) est un standard de télécommunication répandu en France et en Europe. Toutes les informations sont transmises sous forme numérique. Une même ligne téléphonique peut ainsi transporter des signaux de type voix, texte, images et données. Le RNIS permet de conserver les services de télécommunication classiques tout en ouvrant l’accès aux modes de communication futurs (visiophone).

PDM_UserMan_Glossar_fr.fm

Sa vitesse de transmission élevée fait de lui un support idéal pour le télédiagnostic des machines d'emballage.

**Reprise à froid**

Redémarrage d'un programme avec (ré)initialisation de toutes les variables et zones de mémoire (angl.: Cold Restart). La reprise à froid peut être déclenchée manuellement, ou bien automatiquement suite à certains événements.

**Identification (Id)**

Identification d'un objet

**Arbre de renvoi**

cf. Machine d'emballage électronique

**Commentaire**

Textes délimités par des parenthèses et des astérisques (non emboîtables), pour commenter le programme ; le système de programmation ne les interprète pas.

**Communication**

ici : échange de données et d'informations entre configurations

**Données cohérentes**

Ensemble de données formant un tout et ne pouvant être dissociées.

**Langage à contacts (LD)**

Le langage à contacts (angl.: Ladder Diagram - LD) est un langage de programmation conçu pour décrire des réseaux constitués à la fois d'éléments booléens et de composants éléctromécaniques tels que des contacts et des enroulements.

**Lois de courbes**

cf. Génération de consignes

**Court-circuit**

Contact dû à un défaut entre conducteurs sous tension, quand le circuit défectueux ne comporte pas de résistance utile.

**Régulateur de position**

Le régulateur de position ajuste la position effective de sorte que son écart par rapport à la consigne soit aussi réduit que possible. Sortie : Consigne de vitesse.

**Niveau**

(anglais : level) ; état, échelle

**Circuit d’effacement**

Composants destinés à réduire des tensions induites. Les tensions induites sont générées au moment du débranchement de circuits comportant des inductances.

**Adresse logique**

Chaque objet et chaque paramètre de la configuration de commande possède une adresse univoque qui assure qu’il puisse être appelé.

**Masse**

La masse est l’ensemble des parties inactives, interconnectées entre elles, d’une installation dont les parties susceptibles d’être touchées ne peuvent être reliées à une tension dangereuse même en cas de défaut.

**Maître**

Les équipements maîtres commandent la transmission des données sur le bus. Quand il est en possession de l’autorisation d’accès au bus, l’équipement maître peut envoyer des messages sans en avoir reçu la demande auparavant. On appelle les équipements maîtres également utilisateur actif.

**Courbe maîtresse**

Courbe principale ou de base

**MotorController**

Le drive « MotorController » est un équipement électronique comprenant un amplificateur et des boucles de réglage pour la position, la vitesse et le courant d’un servomoteur. La grandeur de sortie, à savoir l’alimentation en puissance du servomoteur, est ajustée en fonction d’une grandeur d’entrée de référence, la consigne de position.

**Variable à plusieurs éléments**

*Variable* de type *tableau* ou *structure* composée de plusieurs *types de données*.

**Couple nominal**

Le couple nominal est le couple permanent exprimé en Nm d’un ensemble servomoteur-amplificateur en régime nominal. Il est inférieur au moment d’inertie correspondant à vitesse 0 du fait de pertes en fonction de la vitesse. Pour une température ambiante de 40°C, la surtempérature du carter moteur sera de 60°C en raison de la constante de temps thermique du moteur.

**Vitesse nominale**

Vitesse utilisable au couple nominal. La vitesse à vide et la vitesse limite mécanique du servomoteur sont plus élevées.

**Filtre réseau**

Dispositif de dérivation de perturbations dans les circuits d'alimentation en puissance en aval du conducteur de protection.

**Réseau**

Ensemble caractérisé par le fait que le contenu du résultat courant n'est pas conservé au-delà de ses limites.

**Top zéro**

Les codeurs incrémentaux émettent un top zéro à chaque rotation. Cette impulsion assure la réinitialisation de la machine.

**OCX**

L'expression **O**LE **C**ontrol E**x**tension désigne des *objets COM* capables de se visualiser eux-mêmes au niveau d'une interface homme-machine. Ces éléments de commande basés COM complètent les éléments de commande courants sous Windows, tels que bouton, case à cocher, etc. Le sigle OCX a été remplacé par la définition plus générale « ActiveX Control » et est tombé en désuétude.

**offline**

non connecté

**OLE**

**O**bject **L**inking and **E**mbedding est un mécanisme basé sur *COM* et conçu pour l'intégration de différents types de document dans un seul document. OLE permet p.ex. d'intégrer une feuille de calcul dans un document texte tout en insérant les fonctionnalités du tableur dans le logiciel de traitement de texte.

**OLE Automation**

OLE Automation définit un mécanisme basé COM qui simplifie la gestion des *objets COM* par les langages script (langages macro d'applications, Java Script, Visual Basic).

**OPC**

Abréviation d'« OLE for Process Control ». La fondation « OPC Foundation » a pour vocation de définir un format standard pour les innombrables interfaces des applications de commande de processus et d'automatisation sur le marché, dans le but d'assurer l'interopérabilité entre les applications de commande, le systèmes et équipements de terrain et les applications bureautiques et

commerciales, et donc de simplifier et accélérer l'intégration de systèmes tout en réduisant le coût. Basé sur les technologies OLE et COM/DCOM de Microsoft, le standard OPC ouvre aux fabricants de matériels informatiques et aux éditeurs de logiciels la possibilité de développer un pilote standard pour l'échange de données avec d'autres systèmes de type HMI (interface homme-machine) ou SCADA (commande de surveillance et acquisition de données).

**Groupes OPC**

Un *groupe OPC* est une unité logique de structuration des *éléments OPC* utilisés par un *client*. Les groupes sont créés par un client et peuvent contenir un ou plusieurs *éléments OPC*. Les *éléments OPC* peuvent faire l'objet d'appels collectifs.

**Eléments OPC**

Les *éléments OPC* représentent les variables de processus. Sont associés aux *éléments OPC*, un identificateur ItemID, une valeur, une information d'état et un horodateur.

**Serveur OPC**

Un *serveur OPC* est un *objet COM* aux *interfaces* conformes à la spécification élaborée par la fondation OPC.

**open**

anglais pour « ouvert »

**Open Control**

La fondation « Open Control Foundation » a pour vocation de développer des concepts de standardisation avancée pour les solutions d'automatisation basées PC. Un projet d'automatisation passe généralement par l'intégration de composantes logicielles et matérielles dans un système tel qu'une commande, un système de visualisation ou différents outils d'ingénierie de projet pour API, bus de terrain ou interfaces homme-machine, ou bien un bus de terrain pour E/S décentralisées. L'Open Control Foundation essaie de faciliter la création de bases de données communes en vue d'une harmonisation des projets d'automatisation, d'une réduction des risques d'erreur et d'une baisse globale du coût d'établissement des projets.

**Opérande**

Elément de langage permettant d'effectuer une opération, p.ex. %IX1

**Opérateur**

Symbole représentant une action nécessaire à l'exécution d'une opération, p.ex. AND

**Optocoupleur**

Liaison optique entre deux systèmes électriquement indépendants.

**PacDrive™**

Le système d’automatisation PacDrive pour PC est optimisé pour l’automatisation intégrale de machines d’emballage. Une intelligence centrale basée sur PC réalise les fonctions API et de commande de mouvement (VarioCam) selon IEC 61131. PacDrive est compatible avec toutes les interfaces homme-machine courantes. Le niveau E/S de l’API est réalisé à l’aide de bus de terrain standards. Le bus temps réel SERCOS assure le raccordement des entraînements asservis les plus dynamiques. L’interface Ethernet intégrée à protocole TCP/IP autorise un télédiagnostic via modem ou internet qui permet d’analyser tous les niveaux du système, jusqu’au courant moteur.

**Variables rémanentes**

Les variables persistantes sont identifiées par le mot clé « PERSISTENT ». A la différence des variables non volatiles, elles conservent leur valeur après un nouveau téléchargement (« online > download »), mais non après l’arrêt et la remise en service de l’API (c'est-à-dire non après la commande « online > reset ») car les variables persistantes ne sont pas sauvegardées dans une mémoire non volatile. Pour conserver leur valeur même en cas d’arrêt intempestif de l’API, les variables persistantes doivent être déclarées VAR RETAIN. Les compteurs d’heures de fonctionnement représentent un exemple concret de « variable non volatile persistante » dans la mesure où ils doivent conserver leur valeur même après une panne secteur.

**PLC**

angl.: Programmable Logic Controller

Voir également « API » à la page 280.

**Plug-In**

Les plug-ins (PI) sont de petits logiciels qui enrichissent les fonctionnalités d’un navigateur pour que celui-ci puisse visualiser des formats de fichier qu’il ne gère pas dans sa version standard.

**UOP**

Spécifiés dans IEC 61131-3, les UOP sont des blocs de type fonction, bloc fonctionnel ou programme qui constituent la structure hiérarchisée d’un API.

**Résolution de position**

La résolution de position indique le nombre d’incréments disponibles par rotation moteur de 360°. Principe de base : La

précision et la qualité de régulation d'un système d'automatisation sont fonction de la résolution de position.

La capacité de résolution des résolveurs varie généralement entre 4096 et 65 000 incréments par tour moteur. Les codeurs SinCos qui équipent le système PacDrive™ offrent même une résolution de 1 000 000 incréments par tour moteur.

**Hors potentiel**

Les potentiels de référence du circuit de commande et du circuit de puissance d'un module E/S hors potentiel sont séparés galvaniquement. Les circuits d'entrée et de sortie n'ont pas le même potentiel de référence. A ne pas confondre avec « à potentiel séparé ».

**A potentiel commun**

Les potentiels de référence du circuit de commande et du circuit de puissance de ce type de modules E/S sont connectés électriquement.

**A potentiel séparé**

Les potentiels de référence du circuit de commande et du circuit de puissance de ce type de module E/S sont séparés galvaniquement, p.ex. au moyen d'optocoupleurs, de relais ou de répétiteurs. Les circuits d'entrée et de sortie peuvent avoir le même potentiel de référence. A ne pas confondre avec « hors potentiel ».

**PROFIBUS**

PROFIBUS est un standard de bus de terrain non propriétaire pour l'automatisation de la fabrication, des processus et des bâtiments. La norme internationale EN 50 170 garantit que ce standard reste authentiquement non propriétaire et donc accessible à tous les acteurs du marché. PROFIBUS assure la bonne communication entre équipements d'origine diverse, sans adaptation préalable des interfaces. PROFIBUS est conçu à la fois pour un échange de données rapide à durée critique et les fonctions de communication complexes et gourmandes en temps de calcul. PROFIBUS est une famille de 3 variantes de bus compatibles entre elles : PROFIBUS-DP, PROFIBUS-FMS et PROFIBUS-PA.

Les raccordements PROFIBUS des produits d'ELAU AG sont basés sur PROFIBUS-DP. PROFIBUS-DP est optimisé pour la communication entre systèmes d'automatisation et équipements périphériques décentralisés au niveau du terrain ; il offre des vitesses élevées tout en réduisant le coût de raccordement. PROFIBUS-DP utilise une structure de bus et une technologie de transmission selon RS-485 ou à fibre optique. La vitesse de transmission est comprise entre 9,6 kBit/s et 12 Mbit/s.

PDM_UserMan_Glossar_fr.fm

**ProgID**

L'identification Program ID est un texte qui se substitue à *CLSID*. Les *ProgID* ont le format « bibliothèque.classe.version » ; à la différence des *CLSID*, ils doivent faire l'objet d'une convention pour être univoques. Exemple d'une identification *ProgID*: « OPC.fabricant.20 ».

**Protocole**

Règles concernant la transmission de données entre les systèmes de calculateurs et les autres équipements. Un protocole définit le paquetage des données, c'est-à-dire le nombre d'octets à envoyer à la fois, le traitement des données transmises avec des erreurs (type de protocole d'erreur), les canaux disponibles (modes demi-duplex ou duplex intégral), l'indication de début et de fin de transmission et le type de compression des paquets de données.

L'organisme de normalisation ISO (International Standards Organization) a établi avec le modèle OSI (Open Systems Interconnection) un modèle de référence pour les transmissions de données entre systèmes électroniques. Un protocole de transmission comporte sept niveaux, conformément au modèle OSI.

**Image de la procédure**

L'image de la procédure renseigne sur l'état de toutes les entrées (= PAE) et toutes les sorties (= PAA) à un moment donné.

**Temps de réaction**

Le temps de réaction est le temps moyen entre une modification au niveau d'une entrée et la modification correspondante au niveau de la sortie.

**Redondance**

Dédoublement d'un équipement. L'unité de réserve se substitue à l'équipement actif en cas de défaut de celui-ci.

**Régulation**

Comparaison entre la valeur de consigne d'une grandeur de procédure et sa valeur effective ; la valeur de consigne est amplifiée [das ist falsch, aber so steht's im dt. Text JR] et ajustée de sorte que la différence entre les deux valeurs soit aussi faible que possible. Contrairement à une fonction de commande, la régulation permet de compenser l'influence d'une grandeur de perturbation. Au niveau du comportement, on distingue généralement la part de la grandeur de référence et celle de la grandeur de perturbation. Un entraînement asservi est régulé en position, en vitesse et en courant.

**Variable rémanente**

Les variables rémanentes peuvent conserver leur valeur au-delà de la durée d'exécution habituelle du programme. Les variables non volatiles et les variables persistantes sont des variables rémanentes.

Voir également « Variables non volatiles (Retain) » à la page 278.

Voir également « Variables rémanentes » à la page 275.

**Remise à zéro**

Remise à l'état initial ; procédure déclenchant le réamorçage d'un « calculateur » (p.ex. pression d'un bouton-pressoir, branchement de la tension de commande).

**Ressource**

En règle générale, ce terme désigne une UC.

**Variables non volatiles (Retain)**

Les variables non volatiles sont identifiées par le mot clé « RETAIN ». Ces variables conservent leur valeur après un arrêt intempestif, comme après un arrêt et un redémarrage normaux (et/ou la commande « online > reset »). A chaque lancement du programme, le système reprend les mêmes valeurs sauvegardées. Le compteur de pièces d'une chaîne de fabrication, qui doit reprendre au même niveau après chaque arrêt, même provoqué par une panne de secteur, est un exemple type de variable non volatile. A la différence des variables persistantes, les variables non volatiles sont réinitialisées [falsch, aber so steht's im dt. Text ; es muss wohl « ne sont pas réinitialisées] lorsque le programme est téléchargée à nouveau. Toutes les autres variables sont réinitialisées, soit avec leurs valeurs initiales soit avec des valeurs par défaut.

**RS-232**

L'interface RS-232-C est une interface série standardisée largement identique à l'interface V.24 (CCITT). Elle est conçue pour un échange de données asynchrone entre le calculateur et un équipement périphérique.

**RS-422**

L'interface RS-422 correspond à RS-232 sur le plan fonctionnel, mais transmet en mode différentiel ce qui lui permet d'offrir des vitesses de transmission élevées sur des distances plus importantes, pour une très faible sensibilité aux perturbations.

**RS-485**

L'interface RS-485 est une interface série standardisée. Du point de vue fonctionnel et électrique, elle est largement identique à

l’interface RS-422, mais elle permet de raccorder plusieurs équipements à un seul câble de connexion (bus).

**SCADA**

Les systèmes de commande de surveillance et d’acquisition de données (Supervisory Control and Data Acquisition ; SCADA) comportent des éléments de commande, d’analyse, de surveillance, de stockage en mémoire et de gestion du flux de données entre les systèmes au niveau du terrain et ceux au niveau du pilotage de l’entreprise. Ils assurent la liaison entre les entrées et sorties décentralisées et les automates programmables, d’une part, et des ordinateurs de bureau, d’autre part.

**Impédance du blindage**

Résistance d’un blindage de câble aux courants alternatifs. L’impédance du blindage est l’une des caractéristiques du câble et figure en général sur la fiche technique fournie par le fabricant.

**Erreur de poursuite**

L’erreur de poursuite est l’écart dynamique entre la consigne de position et la position effective. Cette erreur est fonction de la structure du régulateur et des taux d’amplification dans les différentes boucles de régulation. Les commandes pilotes en vitesse et en courant permettent de réduire l’erreur de poursuite à un minimum.

**Interfaces (OPC)**

Interface COM, identification univoque par code – regroupement de fonctions. Les *objets COM* peuvent avoir une ou plusieurs interfaces. Toute modification postérieure à la publication est interdite.

**Etape**

Nœud d’état d’un programme SFC qui déclenche les instructions relatives à l’*action* prévue à une *étape* déterminée du programme.

**Segment**

Le segment est défini par les terminaisons du câble de bus.

**Serveur**

Un serveur met à disposition des services. Associé au *COM*, un serveur devient *objet COM* dont la fonctionnalité (services) est utilisée par un *client* via ses *interfaces*.

**Servomoteur**

Ce terme désigne désormais un moteur synchrone sans balais à excitation par aimants permanents. Les servomoteurs sont équipés

d’un capteur de position ; ses principales caractéristiques techniques sont un moment d’inertie très faible et sa puissance volumique très élevée.

**Servoamplificateur**

voir MotorController

**Esclave**

Les équipements esclaves sont des périphériques. Les modules E/S illustrent bien cette catégorie d’équipements. Ne pouvant recevoir l’autorisation d’accès au bus, les esclaves émettent des acquittements de message, ou des messages destinés à l’équipement maître qui le leur a demandé. On les appelle également utilisateurs passifs.

**Courbe esclave**

Courbe de superposition. Le lancement d’une courbe esclave suppose qu’une courbe maîtresse soit déjà active.

**Génération d’une valeur de consigne**

Les fonctions de mouvement des axes d’une machine d’emballage électronique sont aujourd’hui réalisées avec les logiciels intelligents du drive PacDrive Controller. Toutes les lois de mouvement spécifiées dans la norme VDI 214 et les lois de mouvement définies par l’utilisateur peuvent être modifiées en ligne.

**Couple crête**

Couple maximal brièvement développé par l’ensemble servomoteur-amplificateur au niveau de l’arbre de sortie du servomoteur.

**Spline Interpolation**

Enchaînement de points spatiaux avec des polynômes du 3e degré.

**API**

Abréviation pour **A**utomate **P**rogrammable **I**ndustriel.

**Fonctions standard**

Ensemble des *fonctions* prescrites dans la norme IEC 61131-3 pour la réalisation des fonctionnalités spécifiques aux API.

**Bloc fonctionnel standard**

Ensemble des *blocs fonctionnels* prescrits dans la norme IEC 61131-3 pour la réalisation des fonctionnalités spécifiques aux API.

**Station**

Equipement qui envoie, reçoit ou amplifie des données sur le bus.

**Commande**

Système signalant uniquement dans le sens entrées-sorties. Ce type de système n'est pas capable de vérifier et corriger les écarts entre les valeurs effectives et les valeurs de consigne.

**Régulateur de courant**

Régulateur maintenant la différence entre la valeur effective et la consigne de courant au niveau le plus bas.

**Littéral structuré (ST)**

Littéral structuré (angl.: Structured Text) est un langage de programmation pour décrire des algorithmes et en commander l'exécution à l'aide des moyens qui caractérisent un langage complexe d'aujourd'hui.

**Variable symbolique**

Variable avec désignateur possédant une *adresse hiérarchique*.

**Syntaxe**

Structure et interaction structurelle entre les éléments linguistiques d'un langage de programmation.

**TCP/IP**

Le sigle « TCP/IP » (abréviation de « transmission control protocol/ internet protocol ») est souvent utilisé pour l'ensemble des protocoles permettant d'interconnecter les ordinateurs de réseaux différents. Le protocole TCP/IP est implanté dans de nombreux réseaux locaux. Il est à la base du réseau mondial Internet.

Le protocole IP gère le transport des données alors que le TCP s'occupe de l'envoi des données à leur destination.

**Télémaintenance**

La télémaintenance permet le raccordement à distance des automates de machines d'emballage via les réseaux de télécommunications (RNIS, GSM) à des fins de maintenance et de réparation ainsi que pour la mise en marche et la modification de machines. La télémaintenance est un atout pour les constructeurs qui réduisent le temps de réaction dans les interventions de maintenance à n'importe quel endroit du globe et garantissent une disponibilité maximale des machines d'emballage.

**Thermocontact de protection**

Interrupteur thermosensible dans l'enroulement du moteur.

**Jeton**

C’est un télégramme autorisant à émettre des messages sur le réseau. Le jeton indique deux états : « libre » et « occupé ». Le jeton circule entre les équipements maîtres.

**Anneau à jeton**

Tous les équipements maîtres reliés physiquement à un bus reçoivent le jeton (token) et le passent au suivant. Ils sont interconnectés en anneau.

**Passage du jeton**

Temps (séquence) s’écoulant entre la réception du jeton et la réception du jeton suivant.

**Tool**

anglais pour « outil »

**Moment d’inertie**

Moment d’inertie du servomoteur exprimé en kgcm2.

**Transition**

Passage d’une étape SFC à la suivante par analyse de la condition de transition.

**Définition de type**

Définition d’un *type de donnée* spécifique à l’utilisateur à partir des *types de données* déjà existants.

**Surcharge de fonctions**

*Fonction* applicable à des entrées pour plusieurs *types de données*.

**UL**

Underwriter Laboratories Inc. (UL) est un organisme de contrôle et de certification autonome aux Etats-Unis. Les produits marqués du sigle UL remplissent les normes de sécurité d’UL.

**Variable**

Mémoire de données contenant des valeurs délimitées par le *type de données* et les informations spécifiées dans la *déclaration* de la variable.

**VarioCam®**

La boîte d’outils en ligne VarioCam® offre un choix étendu de fonctions et notamment de courbes standard. Elle permet d’éviter les modifications et les chargements répétés grâce à un mécanisme de paramétrage du mouvement recherché.

**VxWorks**

Le PacDrive Controller MAx-4 d'ELAU AG utilise le système d'exploitation temps réel VxWorks® de Wind River Systems. VxWorks® est leader du marché des systèmes d'exploitation à temps réel et offre des performances exceptionnelles en matière de comportement temps réel, de solidité et d'efficience de traitement et de stockage.

**Reprise à chaud**

Redémarrage d'un programme à l'endroit où il a été interrompu par une panne de courant (angl.: Hot Restart) pendant que l'API était en mode RUN. Tous les ensembles de données copiés dans les mémoires intermédiaires sont conservées ; l'exécution du programme continue comme si l'interruption n'avait pas eu lieu. Contrairement à la *reprise à chaud depuis le début du programme*, la reprise après interruption ne doit pas dépasser un durée fixée en fonction du processus. Le système d'API doit donc posséder une horloge temps réel permettant au programme de calculer la durée de l'interruption.

**Reprise à chaud depuis le début du programme**

Démarrage du programme comme par *reprise à chaud*, à cette différence près que le programme redémarre depuis le début si l'interruption a duré au-delà d'un intervalle limite. Le programme d'application sait reconnaître cette situation grâce à un indicateur d'état et définit des options par défaut appropriées (angl.: Warm Restart).

**Chien de garde**

(angl. : Watchdog)

Le chien de garde est un dispositif de surveillance cyclique d'équipements, de connexions et de programmes. Un serveur peut demander à un client de lui transmettre régulièrement des signaux. Le serveur contrôle l'arrivée des signaux au moment prévu. L'absence d'un ou de plusieurs signaux serait alors interprétée comme une possible défaillance, soit du client soit de la connexion. Le chien de garde est un circuit matériel ou un protocole logiciel.

**Reprise**

Terme générique pour *reprise à chaud* et *reprise à chaud depuis le début du programme*.

**Windows**

Les logiciels d'exploitation de Microsoft sont massivement plébiscités par les constructeurs de systèmes d'automatisation. Ce phénomène n'est qu'une confirmation de plus de leur tendance, constatée depuis des années, à privilégier les solutions

d’automatisation globales, ouvertes et à échelle modulable basées sur du matériel PC très évolutif. C’est le logiciel qui devient le facteur déterminant des solutions d’automatisation. La technologie 32 bits des systèmes d’exploitation Windows est l’un des moteurs de cette évolution ; de nombreux éditeurs de logiciels industriels lui font confiance. Leur produits englobent en règle générale les interfaces standard pour l’importation de graphiques, pour l’échange de données (DDE, ODBC) et pour l’intégration d’applications (OPC). Ces interfaces représentent un moyen simple pour réaliser les procédures de communication entre logiciels.

Mais ces systèmes d’exploitation standardisés destinés initialement à la bureautique ne sont pas adaptés aux critères de fonctionnement en temps réel qui caractérisent les systèmes d’entraînement. Dans leur version standard, Windows CE, Windows NT et Windows NT Embedded ne conviennent qu’aux applications à « temps réel mou ». Le PacDrive Controller MAx-4 d’ELAU AG est donc basé sur le système d’exploitation à temps réel VxWorks®, qui assure un comportement absolument déterministe. Avec son interface OPC, le PacDrive Controller MAx-4 est facile à intégrer dans un environnement Windows.

**Point commun**

Les modules à point commun ont des circuits d’entrée et de sortie reliés au même point de raccordement. Ce point de raccordement correspond, soit au potentiel (L-) (point commun M) soit au potentiel (L+) (point commun P).

**WWW**

Abréviation de « world wide web », littéralement : toile d’araignée mondiale ; il s’agit d’un système d’information hypertexte et multimédia dans l’internet.

**Circuit intermédiaire**

Tension de puissance redressée et lissée.

**Cycle**

Une répétition d’un programme d’application (appelé périodiquement).

**Temps de cycle**

Temps nécessaire à un *cycle* du programme d’application.

# 15 Annexe

## 15.1 Adresses de contact

**Pour les réparations**

Envoyez les composants qui doivent être réparés ou vérifiés avec le rapport de défaut à l'adresse suivante :

**ELAU AG**
Abt. Kundendienst (Service après-vente)
Postfach 1255
97821 Marktheidenfeld

Hausanschrift (adresse) :
Dillberg 12
97828 Marktheidenfeld

Tél. : +49 (0) 93 91 / 606 - 142
Fax : +49 (0) 93 91 / 606 -340

**Personnel du service après-vente**

Si vous souhaitez un entretien avec l'un de nos collaborateurs ou une intervention sur place :

**ELAU AG**
Dillberg 12
97828 Marktheidenfeld
Tél. : +49 (0) 93 91 / 606 - 0
Fax : +49 (0) 93 91 / 606 -300
eMail : info@elau.de
Internet : www.elau.de

**ELAU, Inc.**
4201 West Wrightwood Avenue
Chicago, Illinois 60639 - USA
Tél. : +1 773 342 8400
Fax : +1 773 342 8404
eMail : sales@elau.com
Internet : www.elau.com

**ELAU SYSTEMS ITALIA S.r.l.**
Via Tosarelli 300
I-40050 Villanova di Castenaso (BO)
Tél. : +39 051 / 7818 70
Fax : +39 051 / 7818 69
eMail : info@elau.it
Internet : www.elau.it

**REMARQUE**

Vous pouvez trouver d'autres adresses de contact sur la page d'accueil de la société ELAU (http://www.elau.de).

PDM_AnhangStd_fr.fm

## 15.2 Documentation supplémentaire

Il est possible de commander chez la société ELAU les manuels et instructions suivants sur le système PacDrive™ :[1]

**Manuel d'étude**

No. d'article 17 13 00 58 - 00x (DE, EN, FR)

**Manuel de programmation**

No. d'article 17 13 00 61 - 00x (DE, EN, FR)

**Mode d'emploi drive « MotorController MC-4 »**

No. d'article 17 13 00 62 - 00x (DE, EN, IT, FR)

**Mode d'emploi CAN L2**

No. d'article 17 13 00 66 - 00x (DE, EN)

**Mode d'emploi PROFIBUS-DP**

No. d'article 17 13 00 67 - 00x (DE, EN)

**Guide d'utilisation Moteur SM**

No. d'article 17 13 00 68 - 00x (DE, EN, IT, FR)

**Mode d'emploi EPAS-4**

No. d'article 17 13 00 70 - 00x (DE, EN)

**Mode d'emploi PacDrive Controller MAx-4**

No. d'article 17 13 00 71 - 00x (DE, EN, IT, FR)

**Manuel utilisateur Serveur OPC**

No. d'article 17 13 00 73 - 00x (DE, EN)

**Mode d'emploi Device Net**

No. d'article 17 13 00 76 - 00x (DE, EN)

**Mode d'emploi Bibliothèques HMI**

No. d'article 17 13 00 77 - 00x (DE, EN)

**Mode d'emploi Module codeur incrémental INC-4**

No. d'article 17 13 00 78 - 00x (DE, EN)

**Mode d'emploi CANopen**

No. d'article 17 13 00 79 - 00x (DE, EN)

**Mode d'emploi VarioCam® Editeur ECAM-4**

No. d'article 17 13 00 80 - 00x (DE, EN)

**Mode d'emploi PacNet Module PN-4**

No. d'article 17 13 00 81 - 00x (DE, EN)

PDM_AnhangStd_fr.fm

1. No. d'article -000 DEutsch (allemand) -001 ENglisch (anglais) -002 ITalienisch (italien) -003 FRanzösisch (français)

**Mode d'emploi Moteur SR**
No. d'article 17 13 00 82 - 00x (DE, EN)

**Mode d'emploi Borne BusTerminal BT-4/DIO1**
No. d'article 17 13 00 83 - 00x (DE, EN)

**Guide d'utilisation TTS**
No. d'article 17 13 00 88 - 00x (DE)

**Manuel utilisateur Optimisation automatique de régulateur**
No. d'article 17 13 00 89 - 00x (DE, EN)

**Guide d'utilisation PacDrive SCL**
No. d'article 17 13 00 93 - 00x (DE)

**Guide d'utilisation PacDrive PS-4 et PacDrive PD-8**
No. d'article 17 13 00 94 - 00x (DE)

**Manuel utilisateur Evaluation Kit**
No. d'article 17 13 00 95 - 00x (DE)

**Guide d'utilisation PacDrive Controller P600**
No. d'article 17 13 00 96 - 00x (DE)

## 15.3 Formations produit

Nous vous offrons de nombreuses formations sur nos produits.

Nos directeurs des séminaires bénéficient d’une expérience de plusieurs années et vous apprennent à profiter d’une manière optimale les multiples possibilités du système PacDrive™.



**REMARQUE**

Pour des informations plus détaillées ainsi que notre programme actuel de séminaire voir la page d'accueil de la société ELAU (www.elau.de).

## 15.4 Modifications

**07 / 1999**

- Adaptation de termes comme p.ex. MAx-4, MC-4 et PacDrive™
- Actualisation des régions de distribution
- Correction de fautes et additifs

**12 / 1999**

- Séparation du manuel de programmation en partie référence et partie utilisateur
- Adaptation à la version actuelle 00.06.00

**01 / 2001**

- Nouveau chapitre « Bases de la norme IEC-61131 »
- Révision du chapitre « Principes de programmation »
- Révision du chapitre « Télémaintenance »
- Nouveau chapitre : « FAQ (Foire Aux Questions) »
- Elargissement du glossaire

**10 / 2003**

- Nouveau chapitre « Réseaux »
- Chapitre « Télémaintenance », élargissement des recommandations sur la choix d'un modem
- Chapitre « Télémaintenance », élargissement de la section « Configuration de l'ordinateur distant » pour Windows 2000
- Elargissement du chapitre « Principes de programmation »
- Reprise des chapitres « Caractéristiques du système », « Diagrammes des temps » et « Identification de version et compatibilité » du Manuel de référence
- Nouveau chapitre « ENI-4 »



**REMARQUE**

La documentation mise à jour et le service des modifications au produit sont disponibles sur la page d'accueil de la société ELAU (http://www.elau.de).

## 15.5 Stichwortverzeichnis

**A**
A potentiel commun 276
A potentiel séparé 276
Action d'entrée en SFC 61
Action de sortie en SFC 61
Action en SFC 61
ActiveX 259
Administration utilisateur 204
Adressage IP 217
Adresse 89
  hiérarchique 268
  logique 272
Adresse MAC 211, 212
Adresses de service 285
Affectation 55
Affectation d'adresse IP 219
Amorçage 261
Anneau à jeton 282
API 280
APIPA 219
Appel indirect d'un bloc fonctionnel 269
Applet 260
Arbre de renvoi 271
ARP 216
AS Interface (ASI) 260
ASCII 260
Automates programmables (AP) 25
Avertissements 12

**B**
Baud 260
Bibliothèques 112
Bloc 261
Bloc de déclaration 264
Bloc fonctionnel 43
Bloc fonctionnel standard 280
Blocs fonctionnels
  en LD 69
  en ST 55
Blocs fonctionnels (FBD) 66
Bobinage 69
Boucle FOR 57
Branchement alternatif 64
Branchement parallèle 64
Bus 261
Bus de terrain 267
Bus des entraînements 260

**C**
Câblage coaxial 210
Câble à fibre optique 211
CAN 261
Caractéristiques 231
Caractéristiques de performance
  EPAS-4 23
  Moteur SM 24
  MotorController MC-4 24
Champs d'application 40
Chien de garde 283
Circuit d'effacement 272
Circuit intermédiaire 284
Classes d'adresses IP 218
Client 262
Client-serveur 204
CLSID 262
Codeur (rotatif) incrémental 269
Codeur absolu 259
COM 262
Commande 281
Commande numérique (CNC) 262
Commentaire 271
Commission Electrotechnique Internationale (IEC) 28
Communication 271
Commutateur 222
Compatibilité 233
Compatibilité électromagnétique (CEM) 266
Concept de domaine 214
Condition de transition 61
Configuration 29
Connecteur de raccordement de bus 261
Connexion à distance 264
Constante de couple 265
Constantes 86
Contact 68

ControlNet 263
Copyrights 2
Couple crête 280
Couple nominal 272
Courbe esclave 280
Courbe maîtresse 272
Court-circuit 271
CSMA/CD 212
CUL 263
Cv 263
Cycle 284

**D**
Dangers
  classification 12
Débit de bauds 261
Débit de données 213
Déclaration 264
  Constantes 38
  Définition 264
  Variables 38
Défaut à la terre 266
Définition de type 282
Delete log file ... 147
DeviceNet 264
DF 268
DHCP 219
Diagramme fonctionnel 268
Diagramme fonctionnel en séquence (SFC) 59, 259
Diagrammes des temps 235
DNS 220
Documentation 286
Domaines 214
Données cohérentes 271

**E**
ELAU AG 2
eMail
  ELAU AG 2
EN 266
ENI-4 247
Enregistrement de trace 129, 152
Enregistreur de messages 146
  Delete log file ... 147
  Filtrage 149
  Load from file ... 147
  Load log file ... 147
  Reset log entries 147
  Save and reset log file 147
  Save as file ... 147
  Save log file 147
Erreur de poursuite 279
Esclave 280
Etape
  active 60
  Définition 279
  IEC 62
Etape active 61
Ethernet 266

**F**
FBD 267, 268
Filtre réseau 273
Fonction 43
Fonctions 90
Fonctions standard 280
Formations 288
Formes de représentation 10
Frein de maintien 268
Fréquence d'échantillonnage 259
Front 267
FTP 215, 267

**G**
Génération d'une valeur de consigne 280
Gestionnaire de surveillance et de recette 129
global 268
Groupe de travail 203
GSM 268

**H**
Hiérarchie des appels 141, 260
Hors potentiel 276
HTML 269
HTTP 215
HUB 221

**I**
Id 269
Identification (Id) 271
Identification d'axe 259
Identification de version 233

IEC
Etape 62
Task 107
IEC « 61131 » 269
Image de la procédure 277
Impédance du blindage 279
Instance 269
Instanciation 46, 269
Instruction CASE 56
Instruction EXIT 59
Instruction IF 56
Instruction REPEAT 59
Instruction RETURN 55
Instruction WHILE 58
Interface homme-machine (HMI) 268
Interfaces 279
Internet 270
ELAU AG 2
Interrupteur de fin de course 266
Inverter Enable 270
IP 215
IP 20 270
IP 65 270
IP 66 270
IP 67 270
IPSec 215

**J**
Jeton 282

**L**
Langage à contacts (LD) 271
Langage en blocs fonctionnels (FBD) 66, 268
Langages
Blocs fonctionnels (FBD) 66
Diagramme fonctionnel en séquence (SFC) 59
Littéral structuré 52
Schéma à contacts (LD) 67
LD comme FBD 69
Le câble à paire torsadée (twisted-pair cabel) 210
Liste d'instructions (IL) 49, 260
Littéral structuré (ST) 52, 281
Load from file ... 147
Load log file ... 147
Login 128
Lois de courbes 271

**M**
Machine d'emballage 19
Machine d'emballage électronique 266
Machine d'état 111
Maître 272
Manuel de programmation 286
Manuels 286
Marquage CE 262
Marques déposées 2
Masse 272
Mise à la terre 266
Mise au point et test 138
Mode 286
Mode intermittent 139
Modes d'emploi 286
Modificateurs en IL 50
Modifications 289
Moment d'inertie 282
MotorController 272
Multitâche 92

**N**
Navigateur 261
NetWare 214
Niveau 271
Numéro de port 216

**O**
Objets COM 262
OCX 273
offline 273
OPC 273
éléments 274
Groupes 274
Serveur 274
serveur 274
open 274
Open Control 274
Opérande 274
Opérateur 50, 274
Opération en ST 53
Optocoupleur 275
Ordre d'accès aux connexions réseau 228

**P**
PacDrive 275
Page d'accueil
ELAU AG 2
Partie déclaration 39
Passage du jeton 282
Passerelle 224
Pathping 228
Peer to Peer 203
Pictogrammes 12
Ping 225
PLC 275
Plug-In 275
Point commun 284
Point d’arrêt 128
Pont 222
POP 215
Potentiel de référence 261
Préfixe 89, 114
Priorité 29, 40, 99, 103
Prise de terre 266
Procédure temps réel 265
PROFIBUS 276
Profil de mouvement 132
ProgID 277
Programme 43
Programme de traitement 99
Protocole 277
Protocoles d’accès 212

**Q**
Qualificatif 63

**R**
Rapport de défaut 297
Redondance 277
Régions de distribution 285
Régulateur de courant 281
Régulateur de position 271
Régulateur de vitesse 265
Régulation 277
Remise à zéro 278
Réparations 285
Répéteur 221
Reprise 283
Reprise à chaud 283
Reprise à chaud depuis le début du programme 283
Reprise à froid 271
Réseau 273
Réseaux 203
Reset 69
Reset log entries 147
Résistance de terminaison 259
Résolution de position 275
Résolution des noms 220
Ressource 29, 278
RNIS 270
Route print 226
Routeur 223
RS -232 278
RS-422 278
RS-485 278

**S**
Saut 64
Save and reset log file 147
Save as file ... 147
Save log file 147
SCADA 279
Schéma à contacts (LD) 67
Sécurité
consignes 12
Segment 279
Segment de bus 261
Séminaires 288
Serveur 279
Service
Personnel du service après-vente 285
Servo
-amplificateur 280
-moteur 279
Set 69
Signal analogique 260
SMTP 215
Socket 217
Sommaire 19
Spline Interpolation 280
Station 281
Structure non reliée à la terre 266
Subnet Mask 217
Supports de transmission 210
Surcharge de fonctions 282
Surveillance 106
Syntaxe 281
Système d'automatisation

Structure 21
Vue d'ensemble du système 22
Système d’exploitation à temps réel 265
Système de bus 261
Système en temps réel 265

**T**
Table de routage 226
Tableau 267
Tâches 29, 99, 103
TCP 215
TCP/IP 214, 281
Technologie décentralisée 264
Télémaintenance 281
Telnet 215
Temps d’accélération 261
Temps de cycle 26, 105, 284
Temps de réaction 277
Thermocontact de protection 281
Token Passing (Passage du jeton) 213
Tool 282
Top zéro 273
Topologie en anneau 207
Topologie en bus 205
Topologie en étoile 206
Topologies 205
Transition 61, 282
Type de donnée
Configuration de commande 93
dérivé 35, 259
élémentaire 32, 266
Notation 33
Types d’instruction en ST 53

**U**
UC 263
UDP 215
UL 282
Unité d’organisation de programme (UOP)
Définition 275
Structure 44
Utilisation conforme aux prescriptions 13

**V**
Variable 282
à un seul élément 265
directement représentée 264
Variable à plusieurs éléments 272
Variable rémanente 278
Variable symbolique 281
Variables 88
Variables non volatiles (Retain) 278
Variables persistantes 275
VarioCam 282
Vitesse nominale 273
VxWorks 283

**W**
Windows 283
WWW 284

## 15.6 Formulaire de rapport de défaut

Ce rapport de défaut est indispensable pour permettre un traitement efficace de la demande.

Adressez votre rapport de défaut à l'agence ELAU dont vous relevez ou à :

**ELAU AG**
Abt. Kundendienst

Dillberg 12
97828 Marktheidenfeld
**Fax : +49 (0) 93 91 / 606 - 340**

Expéditeur :

| | | |
|---|---|---|
| Société : | Localité : | Date : |
| Service : | Nom : | Tél. : |

**Indications concernant le produit défectueux**

Désignation de l'article : .....

Numéro d'article : .....

Numéro de série : .....

Version du logiciel : .....

Code du matériel : .....

Paramètres joints : oui [ ] non [ ]

Programme IEC joint : oui [ ] non [ ]

**Indications concernant la machine sur laquelle est apparu le défaut :**

Fabricant de la machine : .....

Type : .....

Heures de service : .....

Numéro de la machine : .....

Date de mise en service : .....

Fabricant/type de la commande machine :

.....

ELAU_FoStoerung_fr.fm

**Comment se manifeste le défaut :**

.....

.....

.....

**Indications complémentaires :**

| **Apparition du défaut :** | **Causes :** | **Manifestations secondaires :** |
|---|---|---|
| [ ] toujours présent | [ ] inconnu | [ ] problèmes mécaniques |
| [ ] à la mise en service | [ ] erreur de câblage | [ ] défaillance du bloc d'alimentation (24 V) |
| [ ] apparaît sporadiquement | [ ] dommages mécaniques détérioration | [ ] défaillance du drive « Controller » |
| [ ] apparaît au bout de h env. | [ ] humidité dans l'appareil | [ ] défaillance du moteur |
| [ ] apparaît en cas de vibrations | [ ] codeur défectueux | [ ] rupture de câble |
| [ ] dépendant de la température | | [ ] ventilation insuffisante |
| [ ] corps étranger dans l'appareil | | |

L'armoire contient-elle un climatiseur ? O / N[ ]

De tels défauts sont-ils déjà apparus sur le même axe ?
Combien de fois : .....

Les défauts apparaissent-ils toujours à des jours ou à des heures déterminés ?

.....

Autres indications :

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

ELAU_FoSteerung_fr.fm